

Road network analysis with H2Network: Applications of the Spatial database H2GIS

OGRS 2014
Workshop

Authors

Gwendall Petit, Adam Gouge
Nicolas Fortin, Erwan Bocher, Mireille Lecoeuvre

IRSTV – FR CNRS 2488

Contact : firstname.name@ec-nantes.fr

*Otaniemi Campus, Aalto University, Espoo
June 10-13, 2014*

Research context → Belgrand GEBD

Workshop Goal: Demonstrate practical applications of some of the research conducted during the [Belgrand GEBD](#) project

Purpose: Gather together information engineering techniques and expertise gained in various research projects on the city, mobility and the environment

Concrete objectives:

- Facilitate data access by clarifying their existence, use and access privileges
- Enable the archival, referencing and citation of data sets produced or enriched by public research

Outline

1. Introduction
2. OrbisGIS and H2GIS

Coffee break

3. H2Network
4. Use case
5. Conclusion

Introduction

Background

LAuRE law (December 30, 1996)

→ Air and the rational use of energy

“Everyone has the right to breathe air that does not harm their health”

Cities of more than 100 000 inhabitants required to establish “Urban Mobility Plans” (UMP) for transportation, traffic and parking, and reevaluate them every 5 years

→ Necessity of road network analysis becomes clear

Goals

1. Explain graph analysis techniques for creating and evaluating UMPs
2. Illustrate their usefulness in a concrete example

Tools

OrbisGIS - Geographical Information System

With

- **H2** - SQL database (Java)
- **H2GIS** - Spatial extension
- **H2Network** - Network analysis extension

All open source!

Open-source GIS

- Developed at the IRSTV
- Since 2007
- GPLv3
- 100% Java

V4.1 Espoo under active development with a new SQL engine to access and query data

OrbisGIS online



Official website: www.orbisgis.org

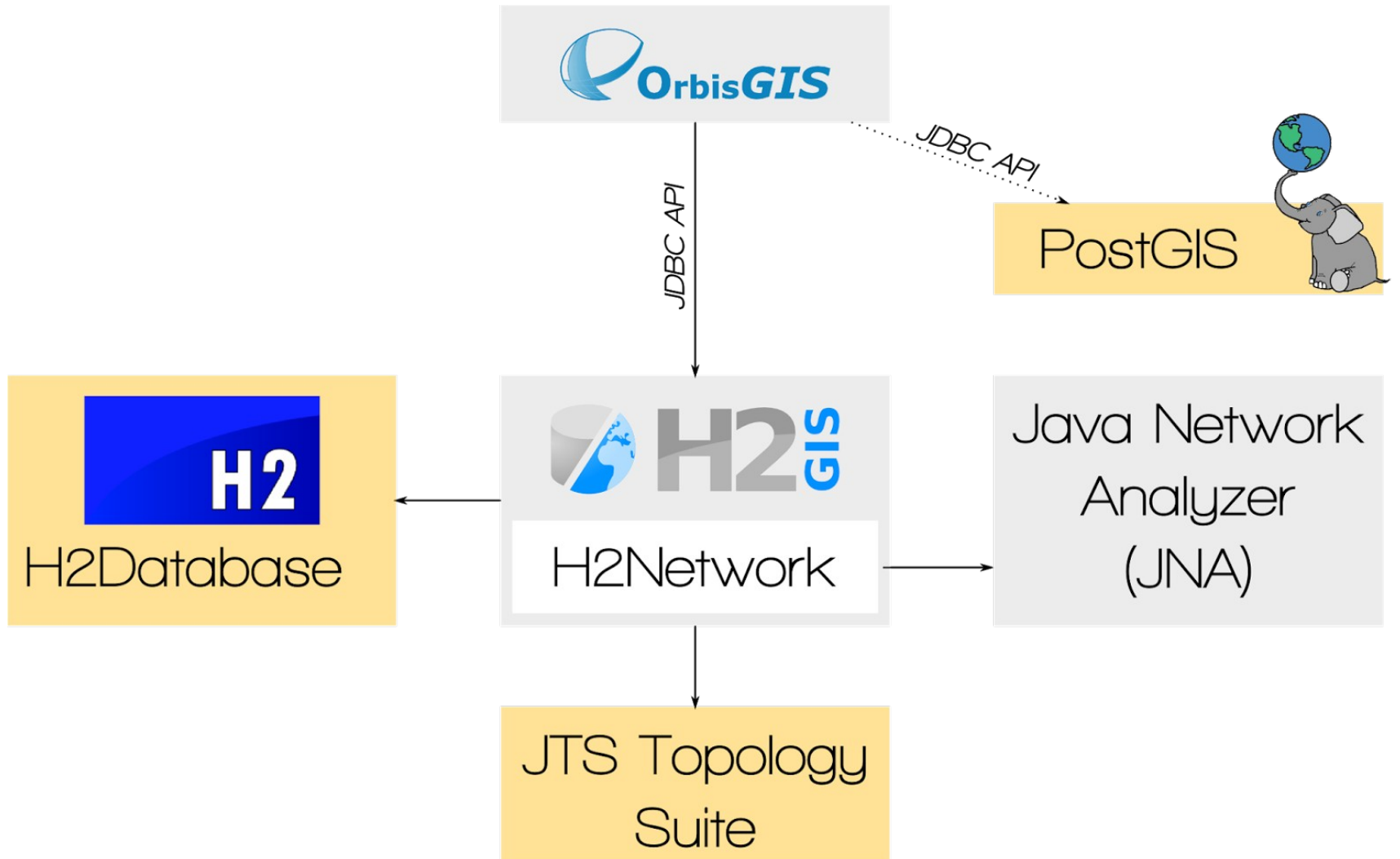
GitHub: <https://github.com/irstv/orbisgis>

Mailing lists (user & developer):
<http://orbisgis.3871844.n2.nabble.com/>

Twitter: <https://twitter.com/OrbisGIS>

Contact us: info@orbisgis.org

The OrbisGIS framework



H2 Database



- A robust and powerful DBMS
- <http://www.h2database.com>
- Open-source ... like PostgreSQL
- 100% Java
- Cross-platform
- No installation (live execution)
- Fully SQL compliant
- Complete documentation
- Connects to a wide range of other DBMSes

A spatial extension of H2 Database

- <http://www.h2gis.org>
- Open-source
- 100% in Java
- Based on the JTS Topology Suite
- Cross-platform & no installation
- Standalone mode available (web interface)
- Implements all SFS functions and additional spatial functions



H2 and H2GIS comprise the new spatial SQL engine for OrbisGIS v4.1 Espoo to

- access
- manage and
- query spatial data

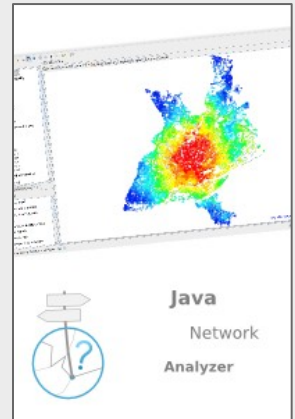
H2GIS passes over 537 unit tests and has an active development community

H2Network

A set of graph analysis functions included in H2GIS

Based on [Java Network Analyzer](#) (JNA) developed at the IRSTV

Uses JGraphT as its graph model



Plan

1. Install OrbisGIS and get used to its user interface
2. Execute some example SQL queries on geographical data
3. Calculate shortest paths and distances in a road network shape file using H2Network
4. Do accessibility calculations for public services such as schools
5. Publish results in a map, using the latest Symbology Encoding specification

Let's get started with OrbisGIS and H2GIS!

Install OrbisGIS

OrbisGIS V4.1 needs at least Java 7

<http://www.java.com>

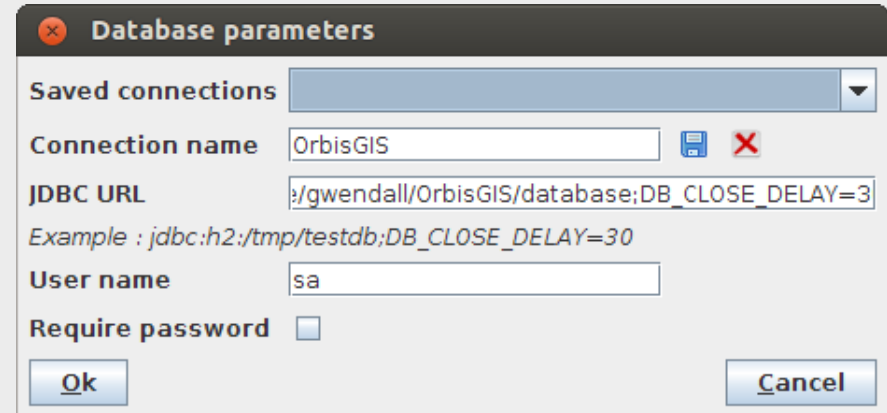
1. Unzip “orbisgis-bin.zip”
2. In the “orbisgis-dist-4.1.0-SNAPSHOT-XX” folder launch OrbisGIS
 - Linux & Mac → `$./orbisgis.sh`
 - Windows → `orbisgis.bat`

Workspace manager

In OrbisGIS, you can organize your work in workspaces

Each of them has a

- path on the machine (or on a USB key)
- connection name
- JDBC URL (local or remote)
- username / password



Workspace manager

At this step it's possible to choose your SQL engine!

You can connect to an H2(GIS) database or to a PostgreSQL/PostGIS database.

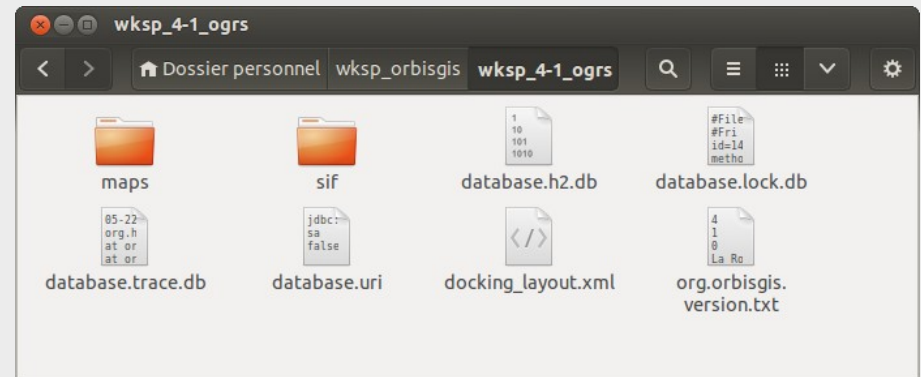
OrbisGIS will use the SQL engine you choose.

Which spatial SQL functions are used depends on this choice

Workspace manager

Accepting the default settings creates a new H2GIS database

Zero configuration!



Note: A “*database.h2.db*” file will be created in your workspace folder

OrbisGIS UI

3 main

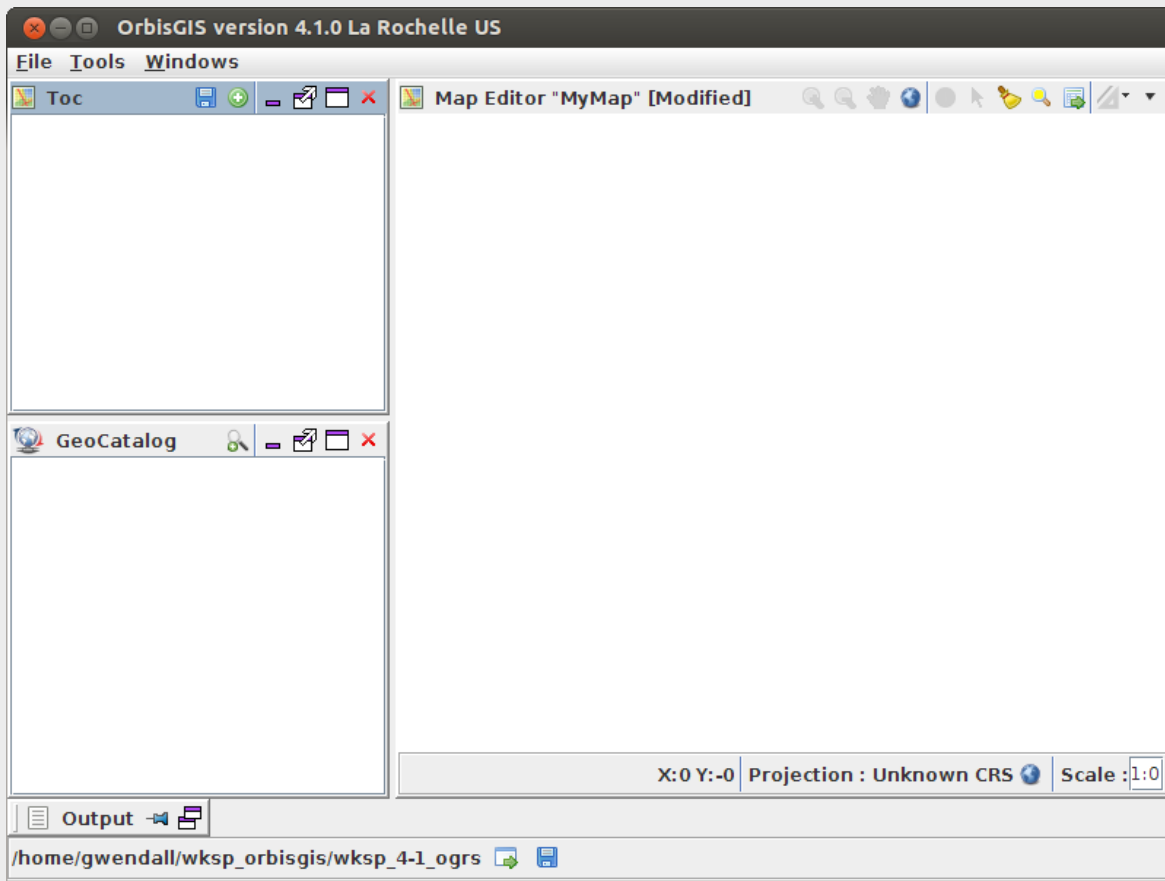
components:

- GeoCatalog
- Table of Contents (TOC)
- Map

→ minimize

→ maximize

→ un/redock

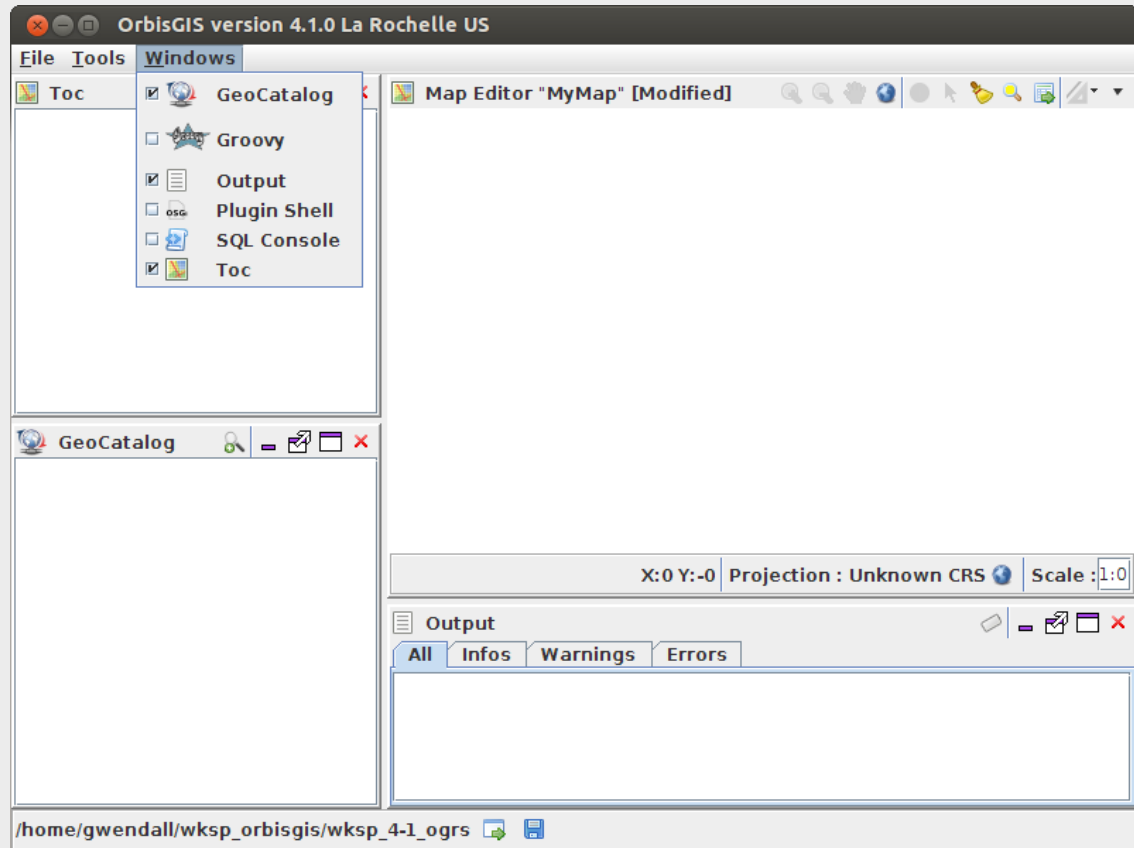


OrbisGIS UI

Other

components:

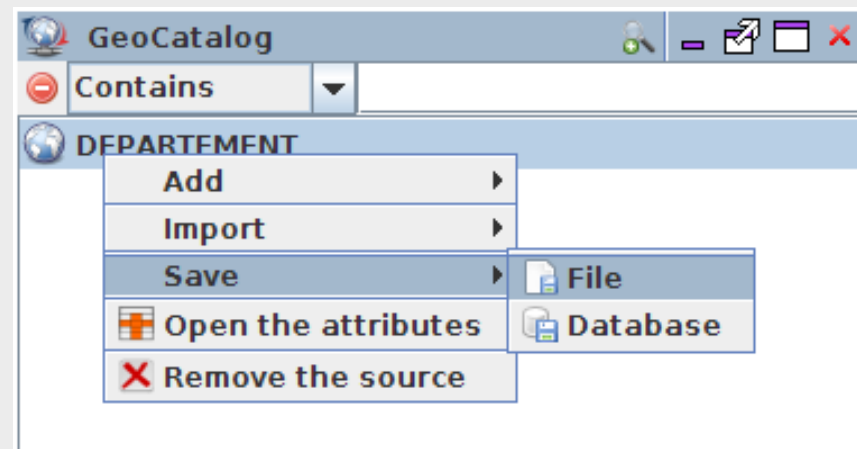
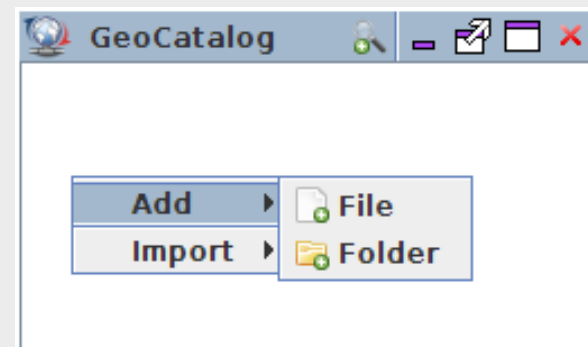
- SQL console
- Groovy console
- Output window
- Plugin manager



Geocatalog

Where you manage
all data (spatial or other)

- Add / Remove
- Import / Export
- Display attributes
- Filter data sources
and system tables



Load data

Open data:

→ OSM (<http://download.geofabrik.de/>)

→ Urban Atlas ([European Environment Agency](#))

All data have been cropped to the Helsinki area

Other data:

→ French departments (from the IGN's [GeoFLA](#) DB)

Loading data

Three choices:

1.Import data into the H2GIS DB

- Read / Write access to imported tables
- Export tables to flat files (i.e., .shp)
- Import time varies with file size

2.Connect to an external DB such as PostGIS (*in progress*)

- Read / Write access

3.Link to a flat file (i.e., .shp/.dbf)

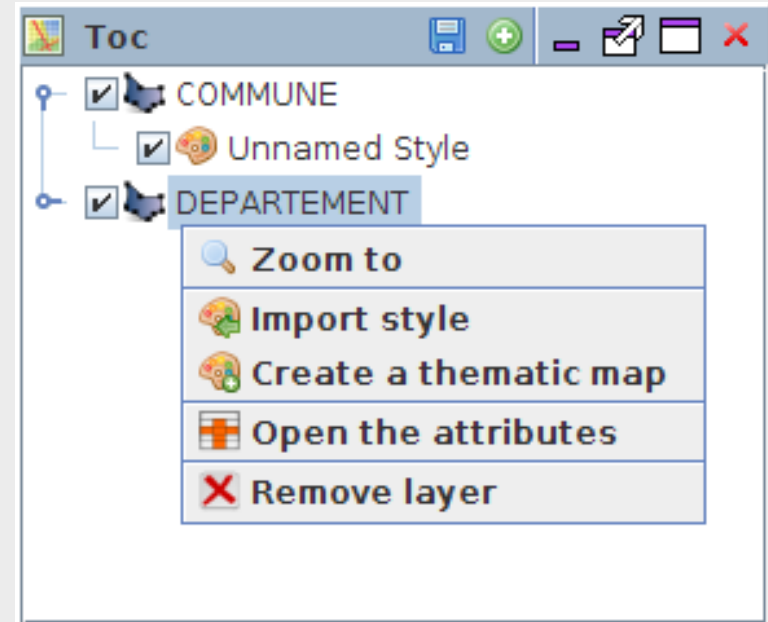
- Read only
- Immediate (no import time)

Display geographic data in map

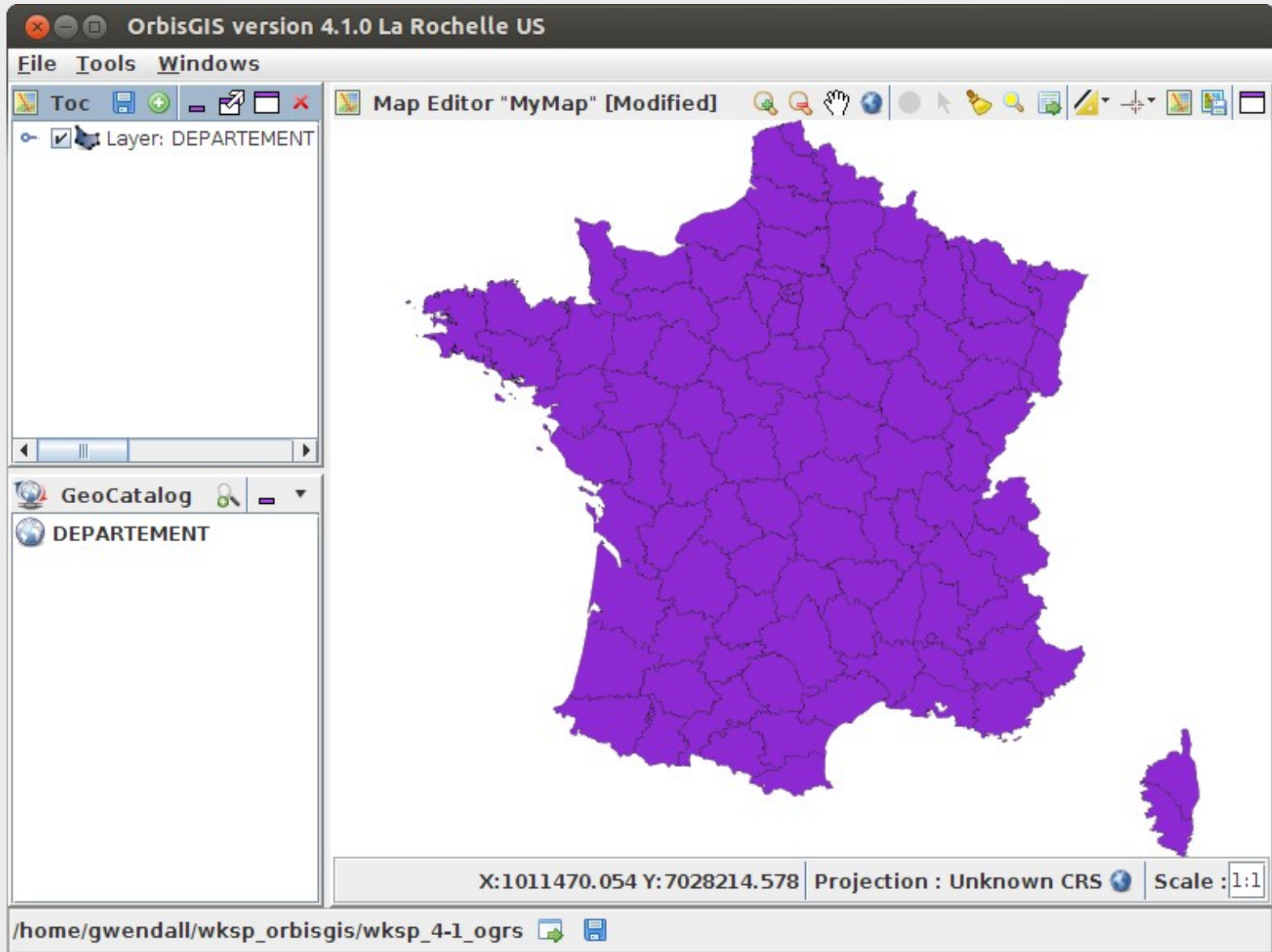
TOC = Table Of Contents

Functionalities:

- Display / Hide layers
- Display attributes
- Order them
- Apply styles (import / create)
- Connect to WMS streams



Navigation tools



Display attributes

The screenshot displays the OrbisGIS interface with a table of department attributes. The table has the following columns: THE_GEOM, ID_GEOFLA, CODE_DEPT, NOM_DEPT, CODE_CHF, NOM_CHF, X_CHF_LIEU, and Y_CHF. The rows list departments from 1 to 21, including AIN, AISNE, ALLIER, ALPES-DE-HAUTE-PROVENCE, HAUTES-ALPES, ALPES-MARITIMES, ARDECHE, ARDENNES, ARIEGE, AUBE, AUDE, AVEYRON, BOUCHES-DU-RHONE, CALVADOS, CANTAL, CHARENTE, CHARENTE-MARITIME, CHER, CORREZE, COTE-D'OR, and COTES-D'ARMOR.

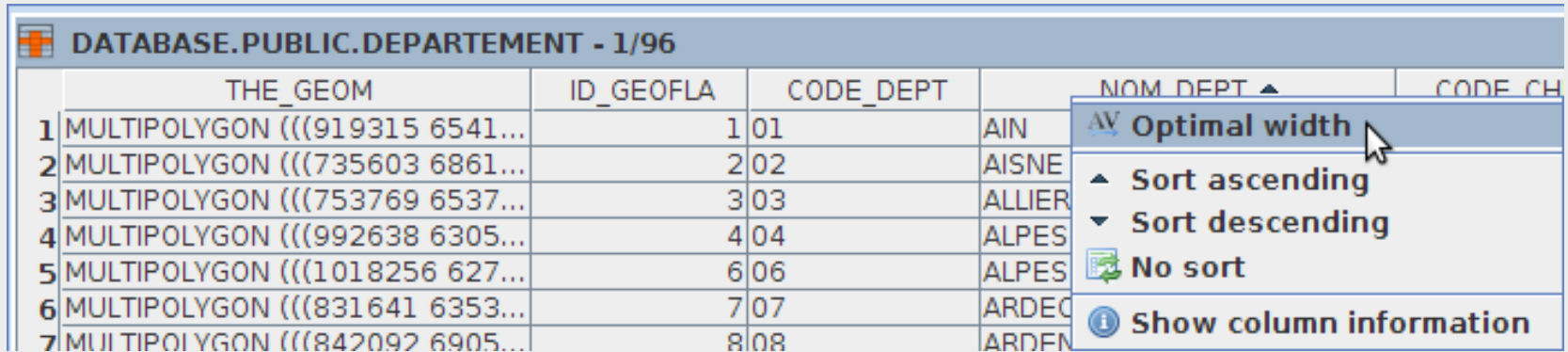
| | THE_GEOM | ID_GEOFLA | CODE_DEPT | NOM_DEPT | CODE_CHF | NOM_CHF | X_CHF_LIEU | Y_CHF |
|----|--------------------------------|-----------|-----------|-------------------------|----------|---------------------|------------|-------|
| 1 | MULTIPOLYGON (((919315 6541... | 1 | 01 | AIN | 053 | BOURG-EN-BRESSE | 8717 | |
| 2 | MULTIPOLYGON (((735603 6861... | 2 | 02 | AISNE | 408 | LAON | 7451 | |
| 3 | MULTIPOLYGON (((753769 6537... | 3 | 03 | ALLIER | 190 | MOULINS | 7254 | |
| 4 | MULTIPOLYGON (((992638 6305... | 4 | 04 | ALPES-DE-HAUTE-PROVENCE | 070 | DIGNE-LES-BAINS | 9590 | |
| 5 | MULTIPOLYGON (((1012913 640... | 5 | 05 | HAUTES-ALPES | 061 | GAP | 9443 | |
| 6 | MULTIPOLYGON (((1018256 627... | 6 | 06 | ALPES-MARITIMES | 088 | NICE | 10439 | |
| 7 | MULTIPOLYGON (((831641 6353... | 7 | 07 | ARDECHE | 186 | PRIVAS | 8266 | |
| 8 | MULTIPOLYGON (((842092 6905... | 8 | 08 | ARDENNES | 105 | CHARLEVILLE-MEZI... | 8239 | |
| 9 | MULTIPOLYGON (((631545 6174... | 9 | 09 | ARIEGE | 122 | FOIX | 5862 | |
| 10 | MULTIPOLYGON (((796594 6759... | 10 | 10 | AUBE | 387 | TROYES | 7799 | |
| 11 | MULTIPOLYGON (((703561 6193... | 11 | 11 | AUDE | 069 | CARCASSONNE | 6472 | |
| 12 | MULTIPOLYGON (((728786 6312... | 12 | 12 | AVEYRON | 202 | RODEZ | 6660 | |
| 13 | MULTIPOLYGON (((917346 6234... | 13 | 13 | BOUCHES-DU-RHONE | 201 | MARSEILLE--1ER-A... | 8934 | |
| 14 | MULTIPOLYGON (((510545 6875... | 14 | 14 | CALVADOS | 118 | CAEN | 4543 | |
| 15 | MULTIPOLYGON (((637122 6391... | 15 | 15 | CANTAL | 014 | AURILLAC | 6557 | |
| 16 | MULTIPOLYGON (((464807 6459... | 16 | 16 | CHARENTE | 015 | ANGOULEME | 4788 | |
| 17 | MULTIPOLYGON (((460931 6449... | 17 | 17 | CHARENTE-MARITIME | 300 | LA ROCHELLE | 3797 | |
| 18 | MULTIPOLYGON (((644784 6591... | 18 | 18 | CHER | 033 | BOURGES | 6541 | |
| 19 | MULTIPOLYGON (((626129 6431... | 19 | 19 | CORREZE | 272 | TULLE | 6037 | |
| 20 | MULTIPOLYGON (((871408 6655... | 20 | 21 | COTE-D'OR | 231 | DIJON | 8542 | |
| 21 | MULTIPOLYGON (((306966 6794... | 21 | 22 | COTES-D'ARMOR | 278 | SAINT-BRIEUC | 2749 | |

Filter with a search engine

Note: Numeric fields are right-aligned

Display attributes

Actions on fields



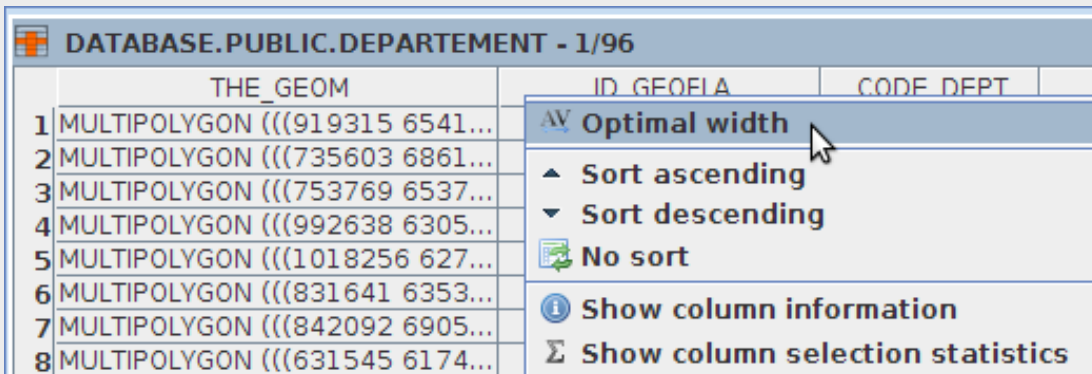
DATABASE.PUBLIC.DEPARTEMENT - 1/96

| | THE_GEOM | ID_GEOFLA | CODE_DEPT | NOM_DEPT ▲ | CODE_CH |
|---|--------------------------------|-----------|-----------|------------|---------|
| 1 | MULTIPOLYGON (((919315 6541... | 1 | 01 | AIN | |
| 2 | MULTIPOLYGON (((735603 6861... | 2 | 02 | AISNE | |
| 3 | MULTIPOLYGON (((753769 6537... | 3 | 03 | ALLIER | |
| 4 | MULTIPOLYGON (((992638 6305... | 4 | 04 | ALPES | |
| 5 | MULTIPOLYGON (((1018256 627... | 6 | 06 | ALPES | |
| 6 | MULTIPOLYGON (((831641 6353... | 7 | 07 | ARDEC | |
| 7 | MULTIPOLYGON (((842092 6905... | 8 | 08 | ARDEN | |

Context menu for 'NOM_DEPT':

- AV Optimal width
- ▲ Sort ascending
- ▼ Sort descending
- 🔄 No sort
- 📘 Show column information

... and on numeric fields



DATABASE.PUBLIC.DEPARTEMENT - 1/96

| | THE_GEOM | ID_GEOFLA | CODE_DEPT |
|---|--------------------------------|-----------|-----------|
| 1 | MULTIPOLYGON (((919315 6541... | | |
| 2 | MULTIPOLYGON (((735603 6861... | | |
| 3 | MULTIPOLYGON (((753769 6537... | | |
| 4 | MULTIPOLYGON (((992638 6305... | | |
| 5 | MULTIPOLYGON (((1018256 627... | | |
| 6 | MULTIPOLYGON (((831641 6353... | | |
| 7 | MULTIPOLYGON (((842092 6905... | | |
| 8 | MULTIPOLYGON (((631545 6174... | | |

Context menu for 'ID_GEOFLA':

- AV Optimal width
- ▲ Sort ascending
- ▼ Sort descending
- 🔄 No sort
- 📘 Show column information
- Σ Show column selection statistics

Table DATABASE.PUBLIC.DEPARTEMENT, statistics of the column ID_GEOFLA.

Row count : 96

Minimum : 1.0

Maximum : 96.0

Sum : 4656.0

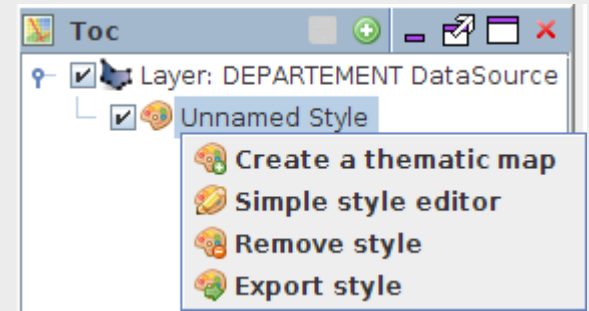
Average : 48.5

Standard deviation : 27.85677655436824

Apply styles

Based on the new OGC Symbology Encoding (SE) standard (<http://www.opengeospatial.org/standards/se>)

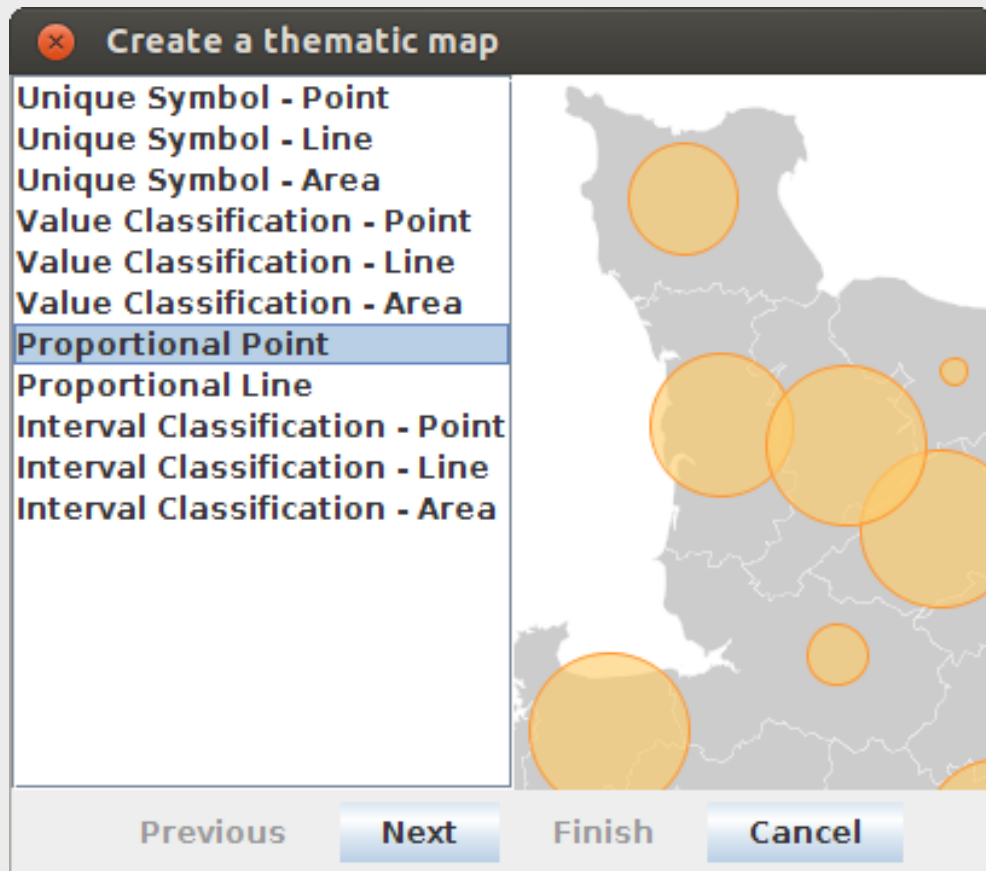
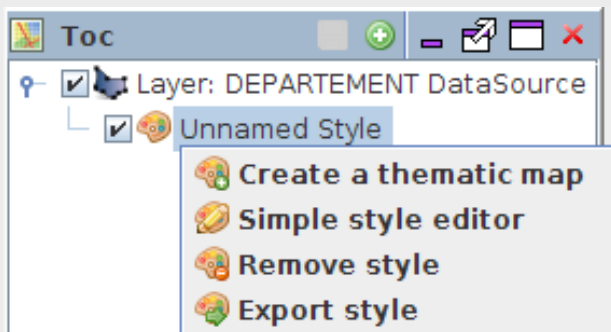
- Progression of SLD
- First implementation in a GIS
- Create / Import / Export .se files (.xml)
- Presented at the UNO in Geneva, today!



```
1 |<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
2 |<ns4:Style xmlns:ns1="http://www.opengis.net/ows/2.0" xmlns:ns2="http://
3 |http://www.opengis.net/ows-context" xmlns:ns4="http://www.opengis.net/s
4 |.net/fes/2.1" xmlns:ns6="http://www.opengis.net/gml" xmlns:ns7="http://
5 |http://www.w3.org/2005/Atom" xmlns:ns9="urn:oasis:names:tc:ciq:xsdchem
6 |.net/sld/1.2" xmlns:ns11="http://www.opengis.net/se/2.0/thematic" xmlns
7 |xmlns:ns13="http://www.w3.org/2001/SMIL20/" xmlns:ns14="http://www.open
8 |gis.net/se/2.0/raster" xmlns:ns16="http://www.w3.org/2001/SMIL20/La
9 |
10 |   <ns4:Name>Urban Atlas</ns4:Name>
11 |   <ns4:Description/>
12 |   <ns4:Rule>
13 |     <ns4:Name>urban_atlas</ns4:Name>
14 |     <ns4:Description/>
15 |     <ns4:AreaSymbolizer uom="urn:ogc:def:uom:se::px" version="2.0.0"
16 |       <ns4:Name>Symbole surfacique</ns4:Name>
17 |       <ns4:SolidFill>
18 |         <ns4:Color>
19 |           <ns4:Recode fallbackValue="#585CA9">
20 |             <ns4:LookupValue>
21 |               <ns5:ValueReference>CODE</ns5:ValueReference>
22 |             </ns4:LookupValue>
23 |             <ns4:MapItem>
24 |               <ns4:Key>11100</ns4:Key>
25 |               <ns4:Value>#8C0000</ns4:Value>
26 |             </ns4:MapItem>
27 |             <ns4:MapItem>
28 |               <ns4:Key>11210</ns4:Key>
29 |               <ns4:Value>#D10000</ns4:Value>
30 |             </ns4:MapItem>
31 |             <ns4:MapItem>
32 |               <ns4:Key>11220</ns4:Key>
33 |               <ns4:Value>#FF0000</ns4:Value>
34 |             </ns4:MapItem>
35 |           </ns4:Recode>
36 |         </ns4:Color>
37 |       </ns4:SolidFill>
38 |     </ns4:AreaSymbolizer>
39 |   </ns4:Rule>
40 | </ns4:Style>
```

Apply styles

Various thematic analyses



Apply styles

Value Classification - Area


Name

General settings

Nonspatial field

Line width unit

Enable border




Fallback symbol

Classification settings







Use the fallback color

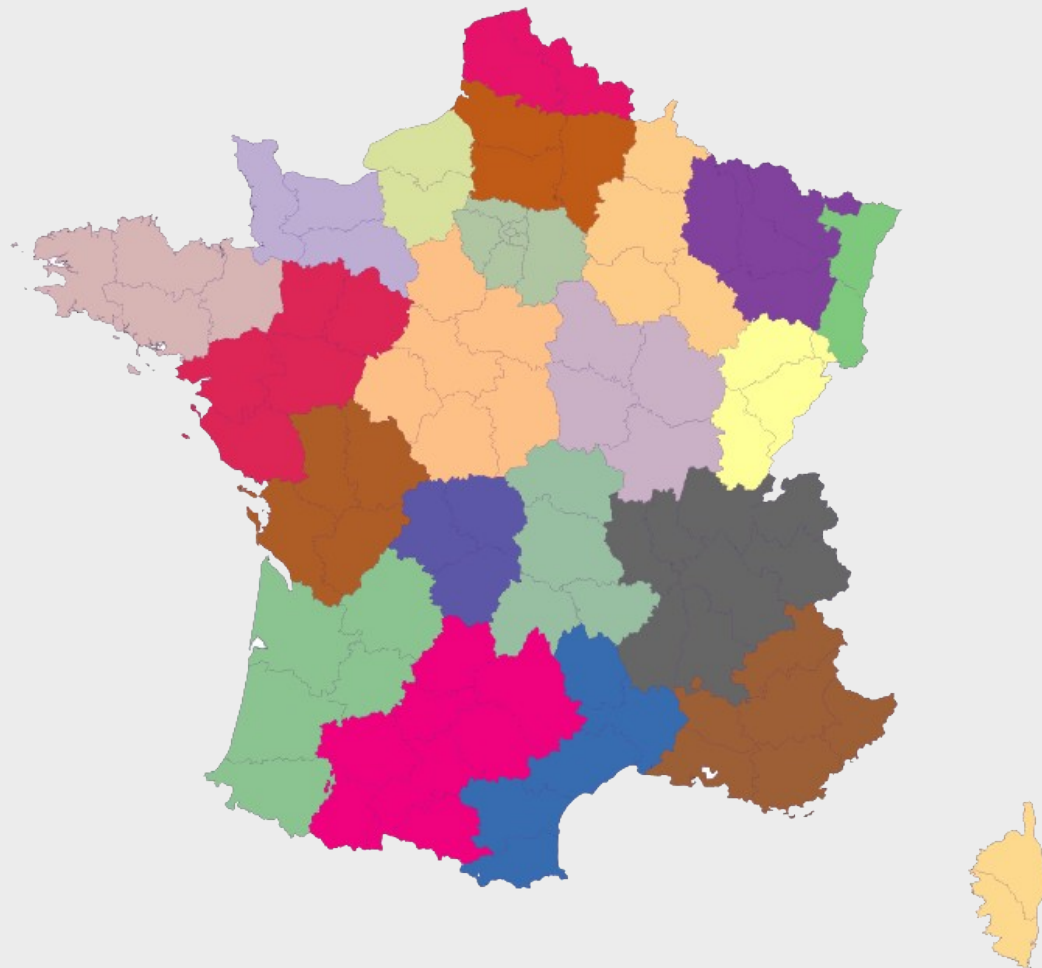
Use a color scheme:

Gradient to



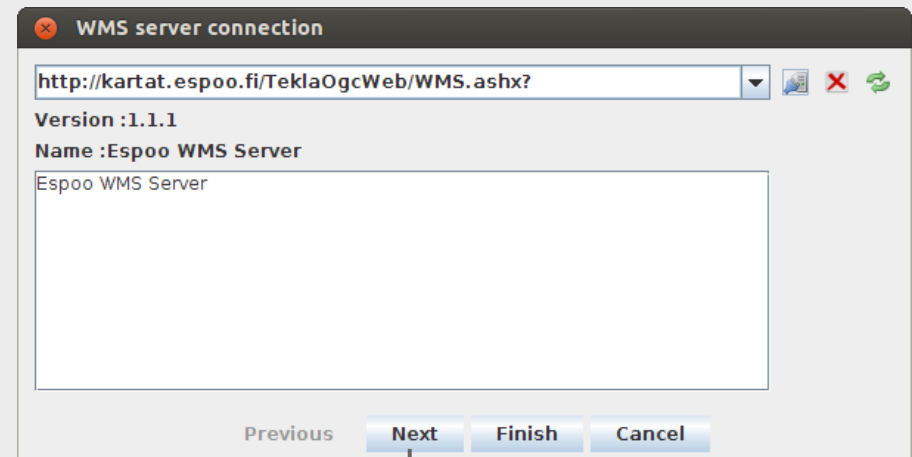
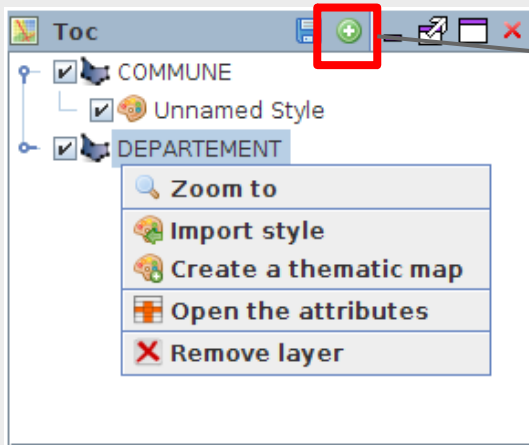
Unique value classification

| Preview | Value |
|--|-------------------|
|  | BASSE-NORMANDIE |
|  | BOURGOGNE |
|  | BRETAGNE |
|  | CENTRE |
|  | CHAMPAGNE-ARDENNE |
|  | CORSE |

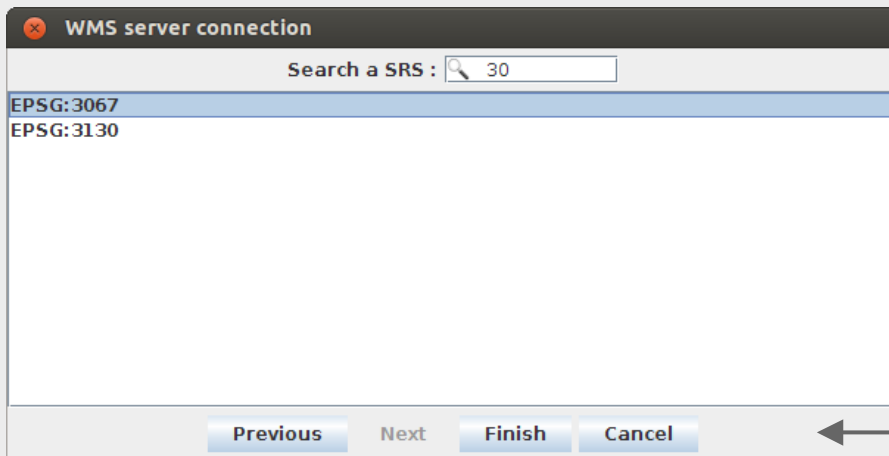


Open a WMS

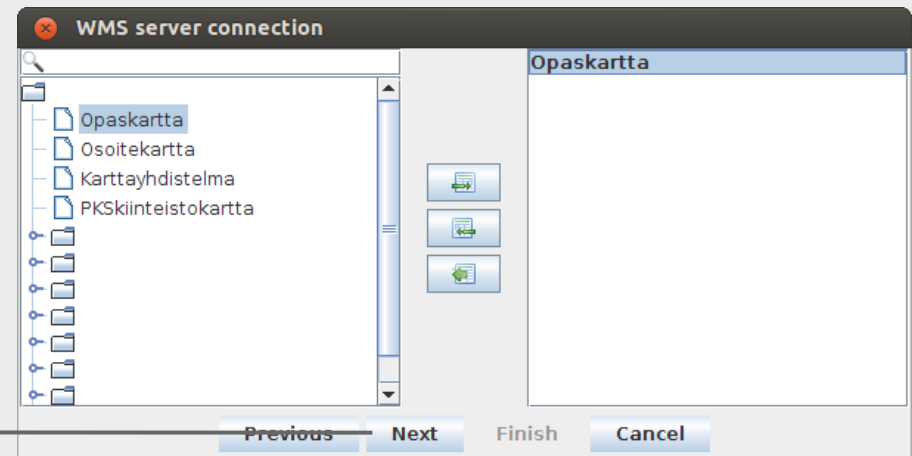
1. WMS URL :
<http://kartat.espoo.fi/TeklaOgcWeb/WMS.ashx?>



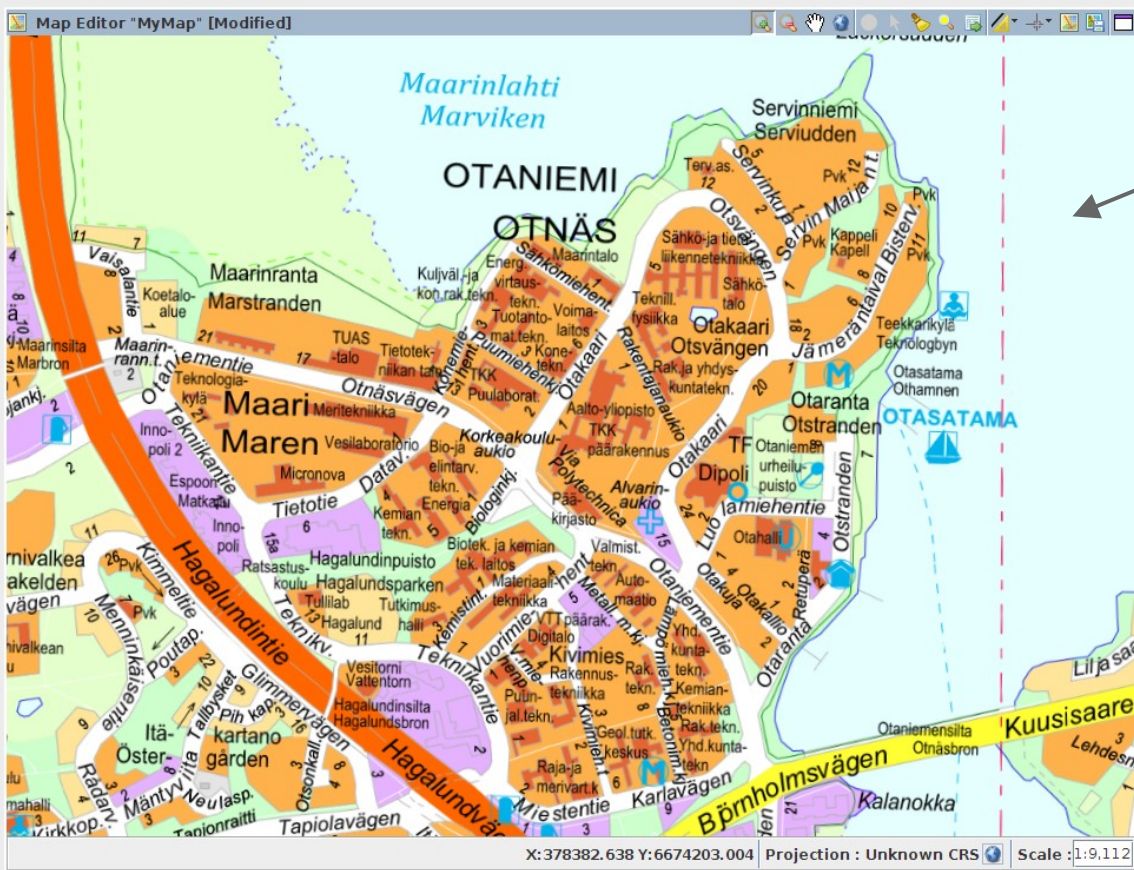
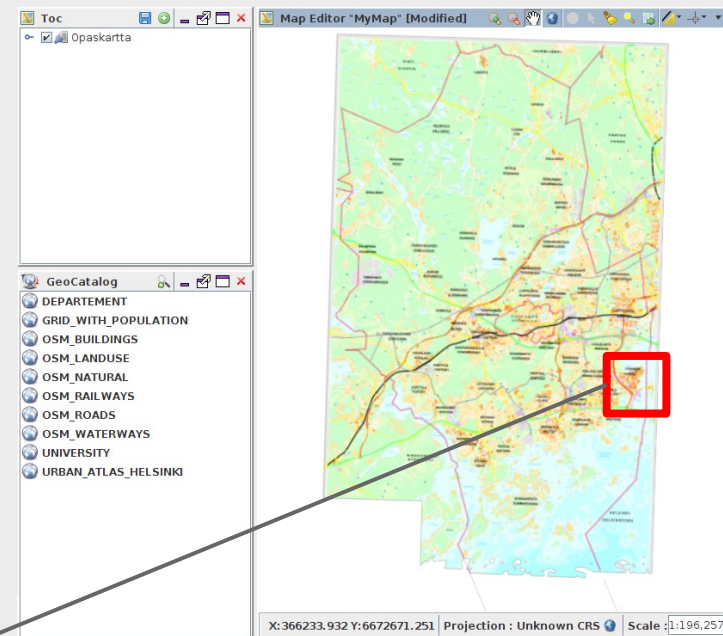
3. Choose CRS ... and click "Finish"



2. Choose layers

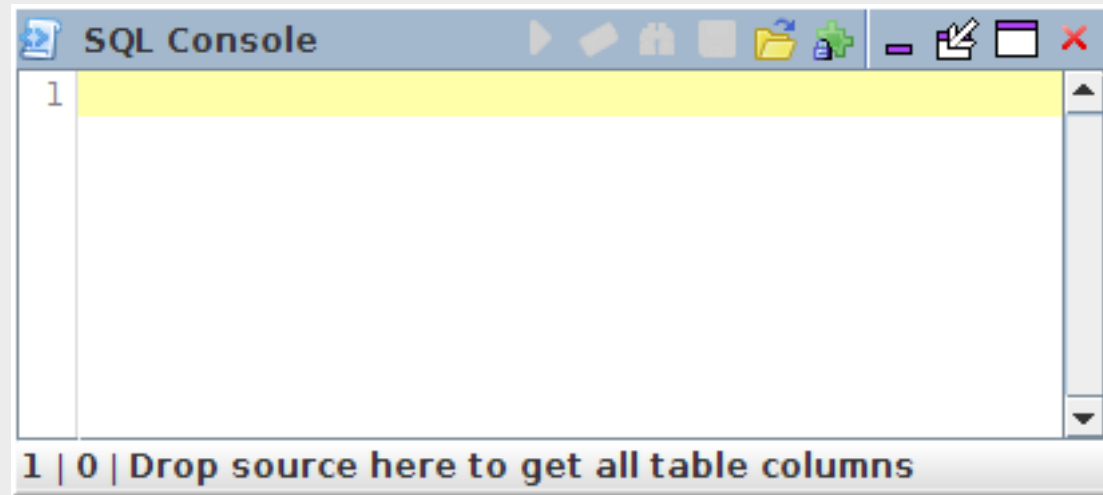


Open a WMS



SQL Console

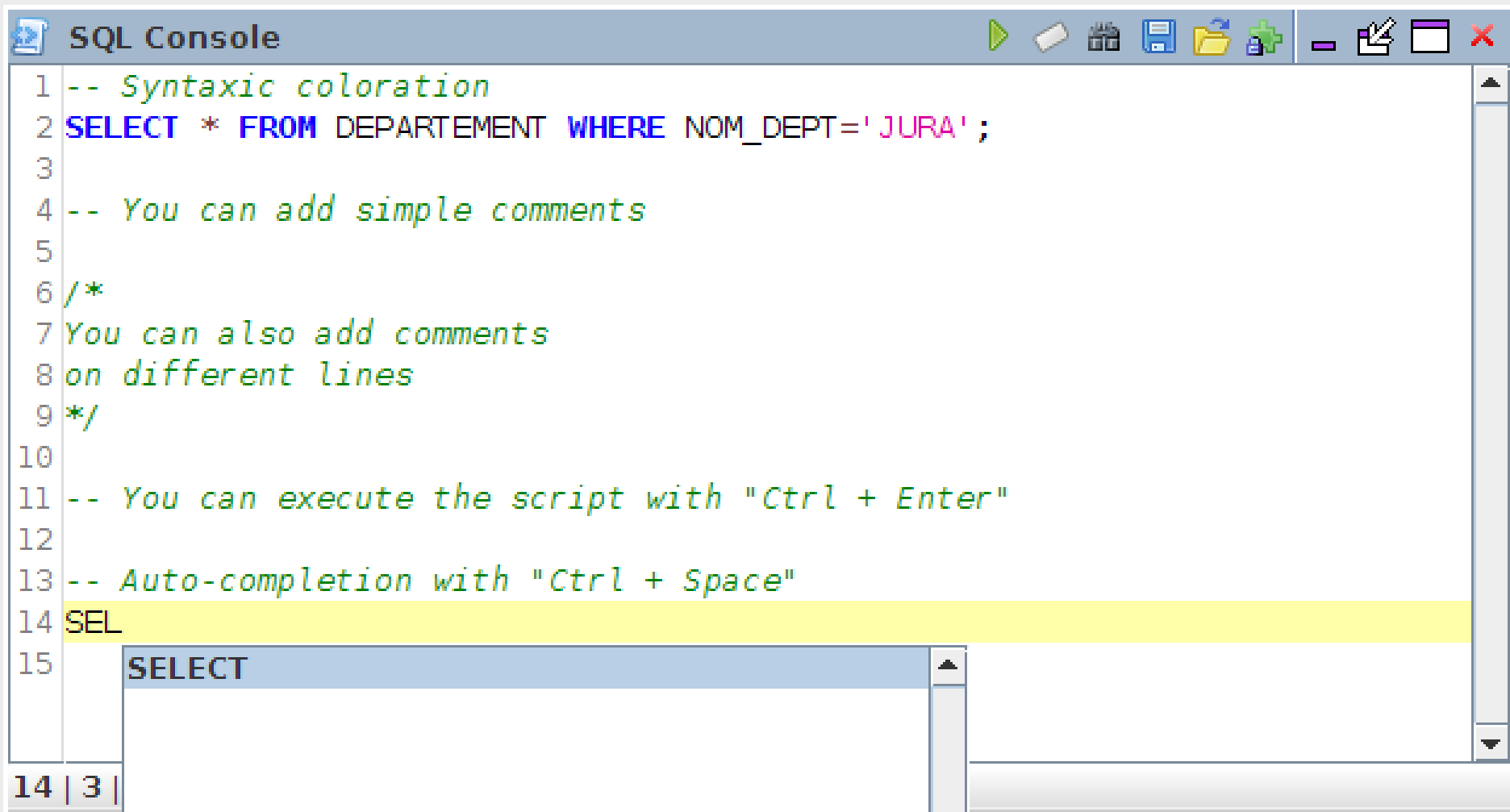
Write / Execute SQL instructions.



Functionalities:

- Execute / Stop query
- Find / Replace
- Save / Open
- Erase
- Display available functions

SQL Console



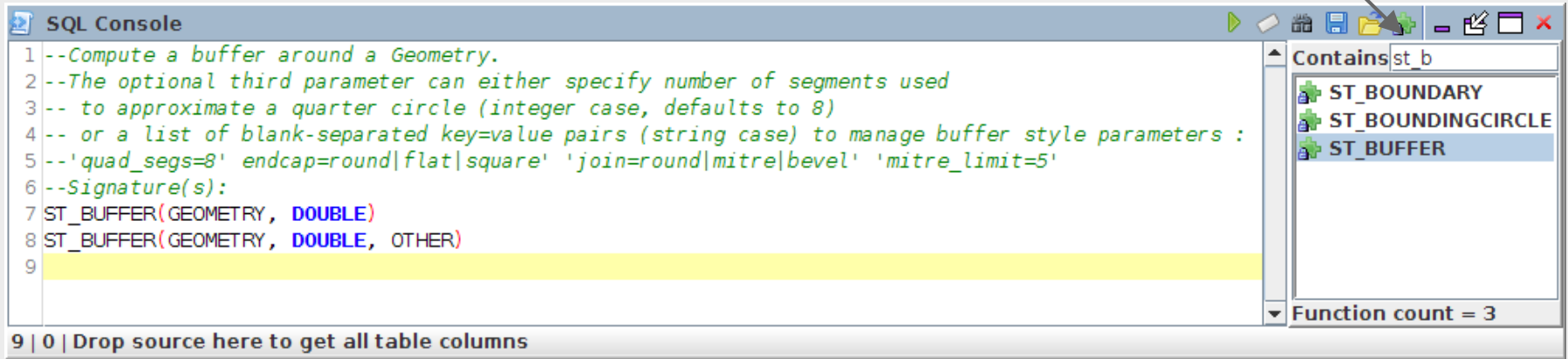
The screenshot shows a window titled "SQL Console" with a toolbar containing icons for execution, erasing, saving, and other functions. The main area contains a SQL script with the following content:

```
1 -- Syntactic coloration
2 SELECT * FROM DEPARTEMENT WHERE NOM_DEPT='JURA';
3
4 -- You can add simple comments
5
6 /*
7 You can also add comments
8 on different lines
9 */
10
11 -- You can execute the script with "Ctrl + Enter"
12
13 -- Auto-completion with "Ctrl + Space"
14 SEL
15
```

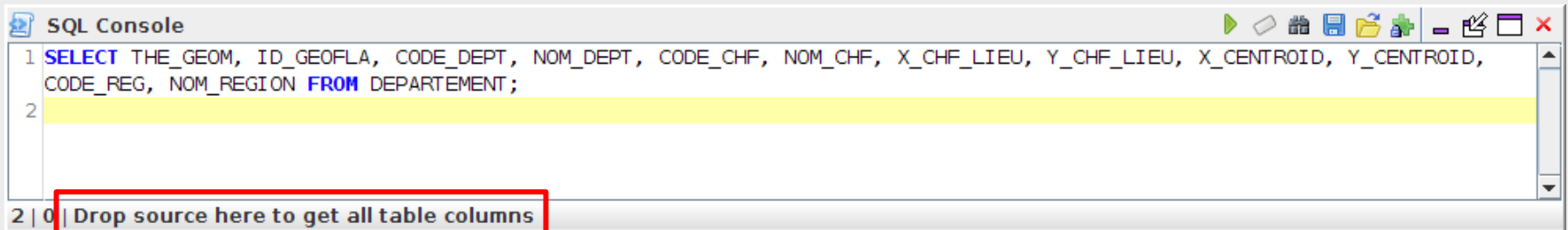
At the bottom left, a status bar shows "14 | 3 |". A dropdown menu is open below line 15, showing the word "SELECT" as a suggestion.

SQL Console

H2GIS functions are shown in a searchable list
→ Drag & drop in console for documentation



→ Drag & drop to obtain all columns of a table

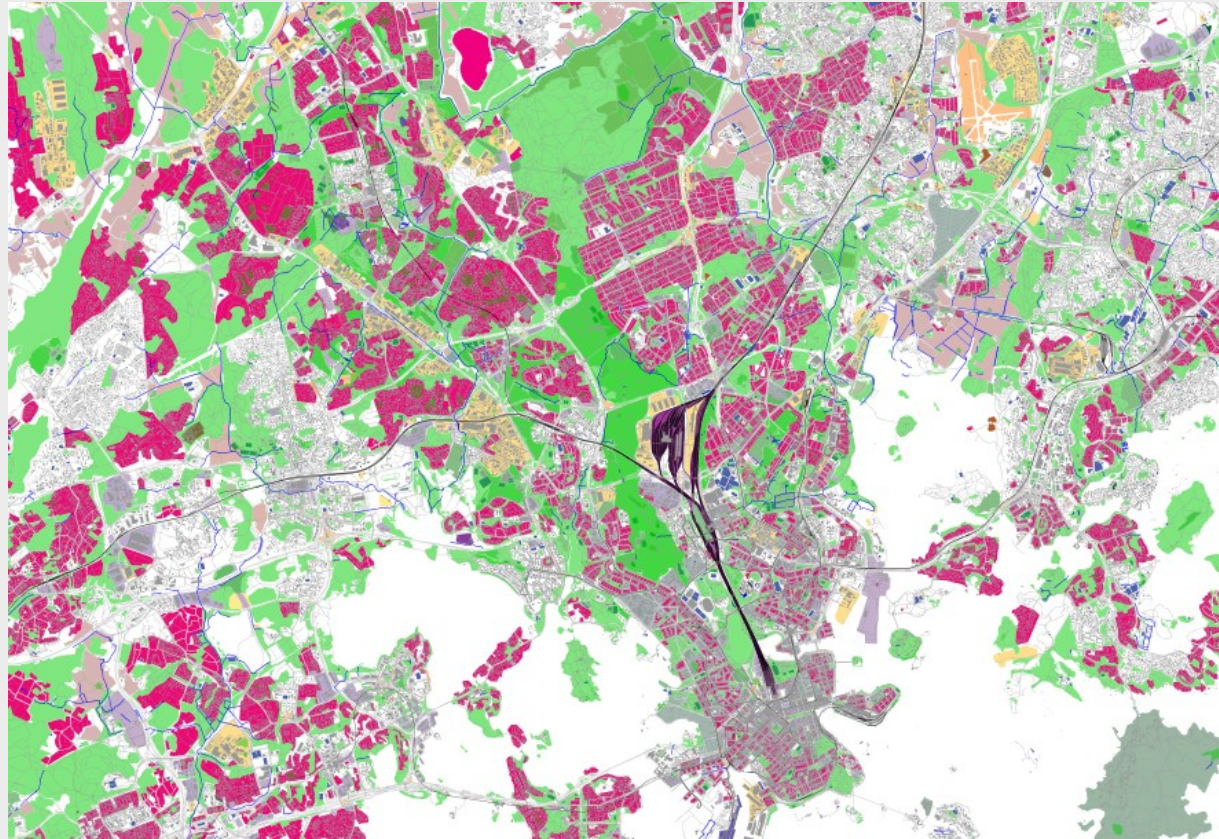


Some SQL exercises



Let's practice (Spatial) SQL with data on Helsinki

- OSM
- Urban Atlas



Some SQL exercises



1. -- Working with the "urban_atlas_helsinki" layer

2.

3. -- "Where condition" on string fields

4. -- Select airports area

5. **SELECT * FROM** urban_atlas_helsinki

6. **WHERE** ITEM ='Airports';

7.

8. -- "Where condition" on numeric fields

9. -- Select small area (less than 200 m²)

10. **CREATE TABLE** small_area **AS SELECT * FROM** urban_atlas_helsinki

11. **WHERE** SHAPE_AREA<200;

12.

13.-- Mixed string and numeric conditions

14. **CREATE TABLE** small_forest **AS SELECT * FROM** URBAN_ATLAS_HELSINKI

15. **WHERE** ITEM='Forests'

16. **AND** SHAPE_AREA<5000;

Some SQL exercises



1. *-- Count the number of forests (code = 30000)*

2. **SELECT** COUNT(*) **AS** nb_forests

3. **FROM** URBAN_ATLAS_HELSINKI

4. **WHERE** CODE='30000';

5. --> answer = 899

6.

7. *-- Use "Group by" and "Order by" operators*

8. **SELECT** CODE, ITEM, SUM(SHAPE_LEN) **AS** SHAPE_LEN, SUM(SHAPE_AREA) **AS**
SHAPE_AREA

9. **FROM** URBAN_ATLAS_HELSINKI

10. **GROUP BY** CODE, ITEM

11. **ORDER BY** CODE **ASC**;

Some SQL exercises

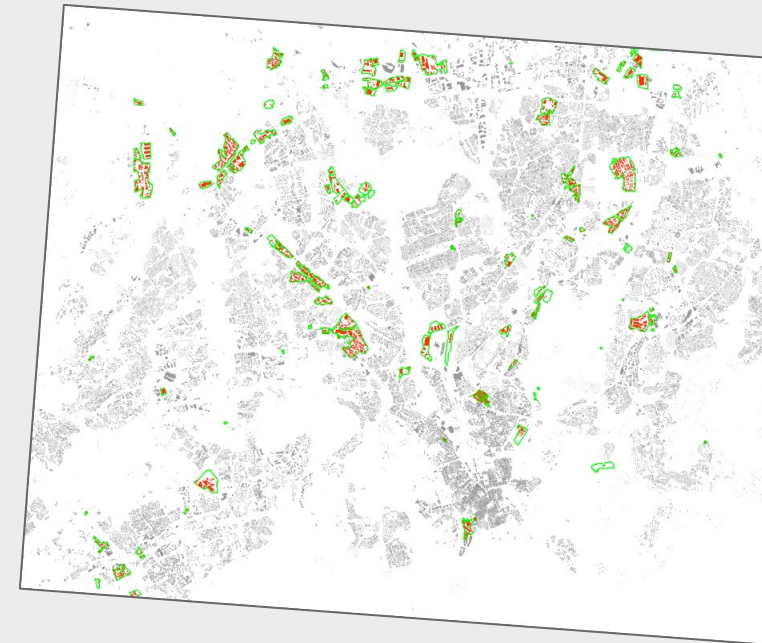


1. -- Create spatial indexes (for speed) and primary keys
- 2.
3. -- On OSM_BUILDINGS
4. -- Create spatial index
5. **CREATE SPATIAL INDEX ON** OSM_BUILDINGS (the_geom);
6. -- Make the future primary key field non-nullable
7. **ALTER TABLE** OSM_BUILDINGS **ALTER COLUMN** OSM_ID **SET NOT NULL;**
8. -- Create the primary key
9. **CREATE PRIMARY KEY ON** OSM_BUILDINGS (OSM_ID);
- 10.
- 11.-- On OSM_ROADS
12. **CREATE SPATIAL INDEX ON** OSM_ROADS (the_geom);
13. **ALTER TABLE** OSM_ROADS **ALTER COLUMN** OSM_ID **SET NOT NULL;**
14. **CREATE PRIMARY KEY ON** OSM_ROADS (OSM_ID);

Some SQL exercises



1. -- Use spatial function in the "where" condition
2. -- Select buildings that are more than 1000 square meters
3. **CREATE TABLE** big_building **AS SELECT** *
4. **FROM** OSM_BUILDINGS
5. **WHERE** ST_AREA(the_geom)>1000;
- 6.
7. -- Select buildings that intersects an industrial zone
8. **CREATE TABLE** building_indus **AS SELECT** a.*
9. **FROM** OSM_BUILDINGS a, OSM_LANDUSE b
10. **WHERE** a.the_geom && b.the_geom
11. **AND** ST_INTERSECTS(a.the_geom, b.the_geom)
12. **AND** b.TYPE_OBJ='industrial';



Some SQL exercises



1. -- Select the intersection between buildings and roads

2. **CREATE TABLE** building_roads **AS SELECT** ST_INTERSECTION(a.the_geom, b.the_geom) **as**
the_geom, OSM_ID, NAME, TYPE_OBJ

3. **FROM** OSM_BUILDINGS a, OSM_ROADS b

4. **WHERE** a.the_geom && b.the_geom

5. **AND** ST_INTERSECTS(a.the_geom, b.the_geom);

6.

7. -- Count the number of lines that are more than 100 meters long

8. **SELECT** COUNT(*)

9. **FROM** building_roads

10. **WHERE** ST_DIMENSION(the_geom)=1 **AND** ST_LENGTH(the_geom)>100;

11.--> answer = 107

Some SQL exercises



1. -- Select buildings that are less than 100 meters far from a motorway

2. -- V1 - in two steps (15.177s)

3. **CREATE TABLE** buffer_area **AS SELECT** ST_UNION(ST_ACCUM(ST_BUFFER(the_geom,
100))) **AS** the_geom

4. **FROM** OSM_ROADS

5. **WHERE** TYPE_OBJ='motorway';

6. **CREATE TABLE** building_in_buffer **AS SELECT** a.*

7. **FROM** OSM_BUILDINGS a, buffer_area b

8. **WHERE** a.the_geom && b.the_geom

9. **AND** ST_INTERSECTS(a.the_geom, b.the_geom);



Some SQL exercises



1. -- Select buildings that are less than 100 meters far from a motorway

2. -- optimized way (0.845s)

3. **CREATE TABLE** buffer_area **AS SELECT** ST_BUFFER(the_geom, 100) **AS** the_geom

4. **FROM** OSM_ROADS

5. **WHERE** TYPE_OBJ='motorway';

6. **CREATE TABLE** building_in_buffer **AS SELECT DISTINCT** a.*

7. **FROM** OSM_BUILDINGS a, buffer_area b

8. **WHERE** a.the_geom && b.the_geom

9. **AND** ST_INTERSECTS(a.the_geom, b.the_geom);

10.-- V2 - in one instruction (1.179s)

11.**CREATE TABLE** building_next_motorway **AS SELECT DISTINCT** a.*

12. **FROM** OSM_BUILDINGS a, OSM_ROADS b

13. **WHERE** a.the_geom && ST_BUFFER(b.the_geom, 100)

14. **AND** ST_INTERSECTS(a.the_geom, ST_BUFFER(b.the_geom, 100))

15. **AND** b.TYPE_OBJ='motorway';

And now H2Network!

Why H2Network ?

- Network analysis in a GIS (OrbisGIS)
- Network analysis in a DBMS (H2)
- Compatibility with PostGIS

H2Network

Created to perform network analysis in OrbisGIS:

- Produce Node and Edge tables from geographical data
- Distances (Point to point)
- Distance matrices (N points to M points)
- Shortest paths
- Shortest path trees (optionally limited by radius)
- Accessibility analysis
- Betweenness centrality
- Closeness centrality
- Strahler stream order

Graph model choices

Java libraries:

- Graphhopper
 - API not very stable at the time
 - Somewhat difficult to extend
- JUNG
 - Community no longer active
- GraphStream
- Grph
- JGraphT

Non-Java

- PGRouting (C/C++)
 - Requires a PostgreSQL DB

Graph model choice: JGraphT

- Java
- Stable API (project started in 2003)
- Large community
- Many algorithms already implemented
- IRSTV had previously developed tools in JGraphT

H2Network, based on JNA

All algorithms implemented in the Java Network Analyzer library

Node and Edge tables are produced using SQL

H2Network = the bridge between H2GIS and JNA

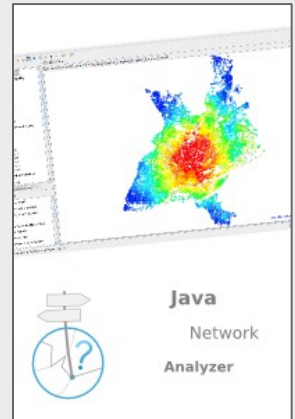
JNA algorithms

Implemented on JGraphT graphs

Graph types

- oriented (directed, reversed, undirected)
- weighted (optional)

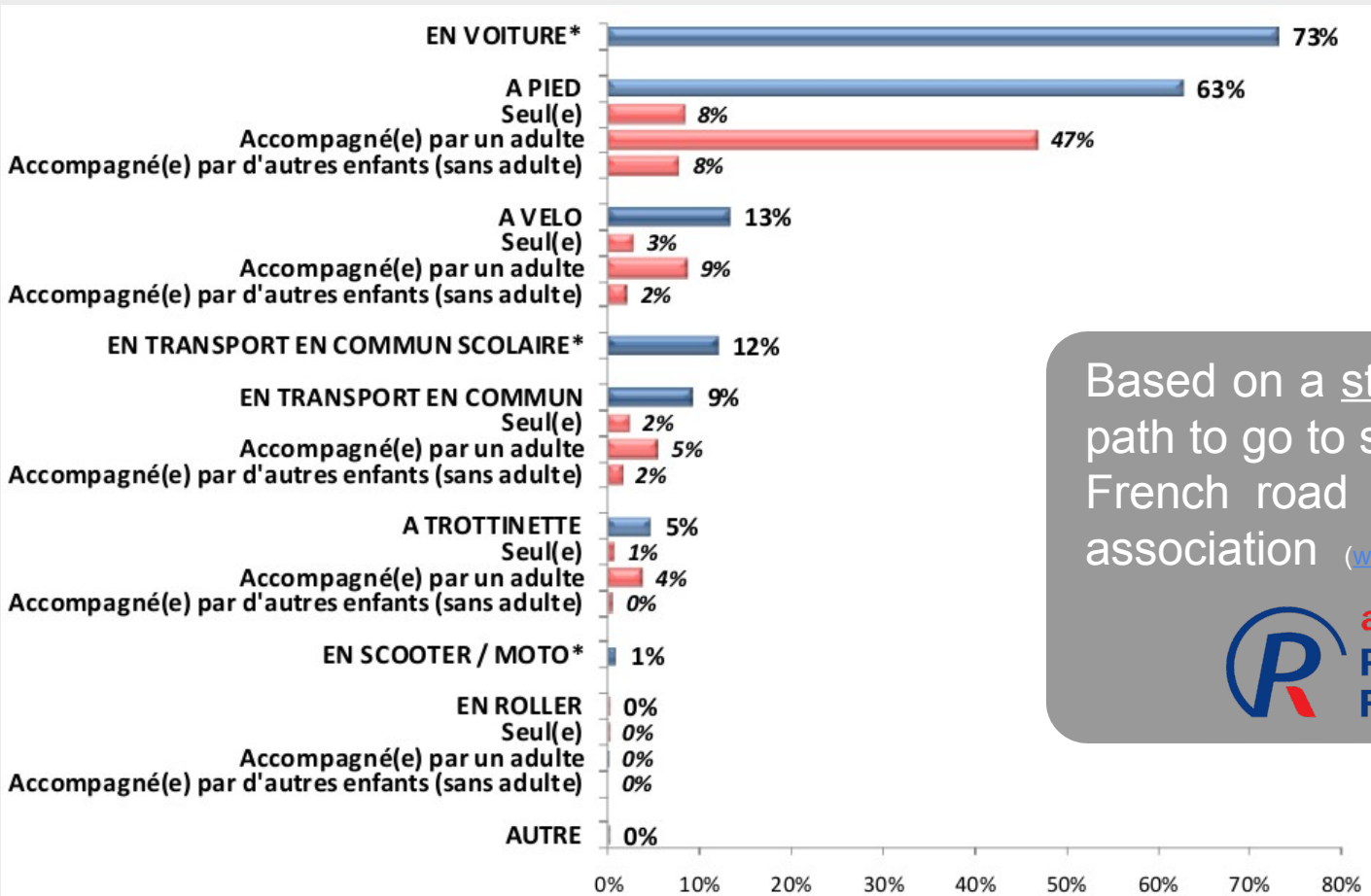
<https://github.com/irstv/java-network-analyzer>



Case study

Case study: time to access schools

In France, 73% of children go to school by car



Based on a study focused on the path to go to school, made by the French road security prevention association (www.preventionroutiere.asso.fr)



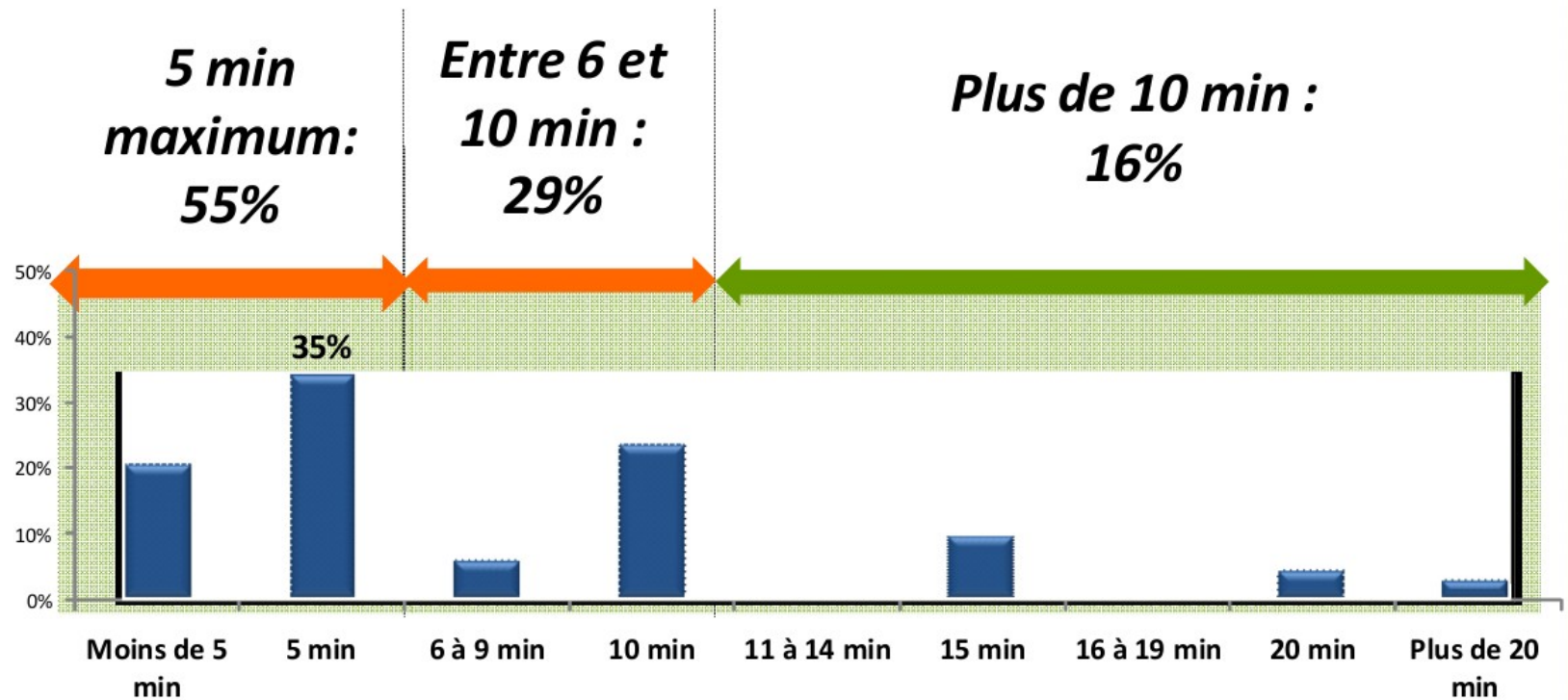
* = with an adult

Schema extracted from http://www.preventionroutiere.asso.fr/content/download/4418/42864/version/2/file/Rue+Tom+et+Lila_resultats+enqu%C3%AAte_13.09.2011.pdf

Case study: time to access schools

In France, 84% of children are less than 10 min from school

En moyenne, quelle est la durée du trajet domicile / école ?



Schema extracted from http://www.preventionroutiere.asso.fr/content/download/4418/42864/version/2/file/Rue+Tom+et+Lila_resultats+enqu%C3%AAte_13.09.2011.pdf

Case study: time to access schools

Context: Reevaluating a UMP

- Every 5 years
- Examine various scenarios

 What is the impact of modifying a UMP on accessibility to services?

Scenarios

- S1** 2010 OSM Reference situation

- S2** S1 + Eric Tabarly Bridge joining two major neighborhoods of Nantes (Ile de Nantes / Malakoff), naturally separated by the Loire river

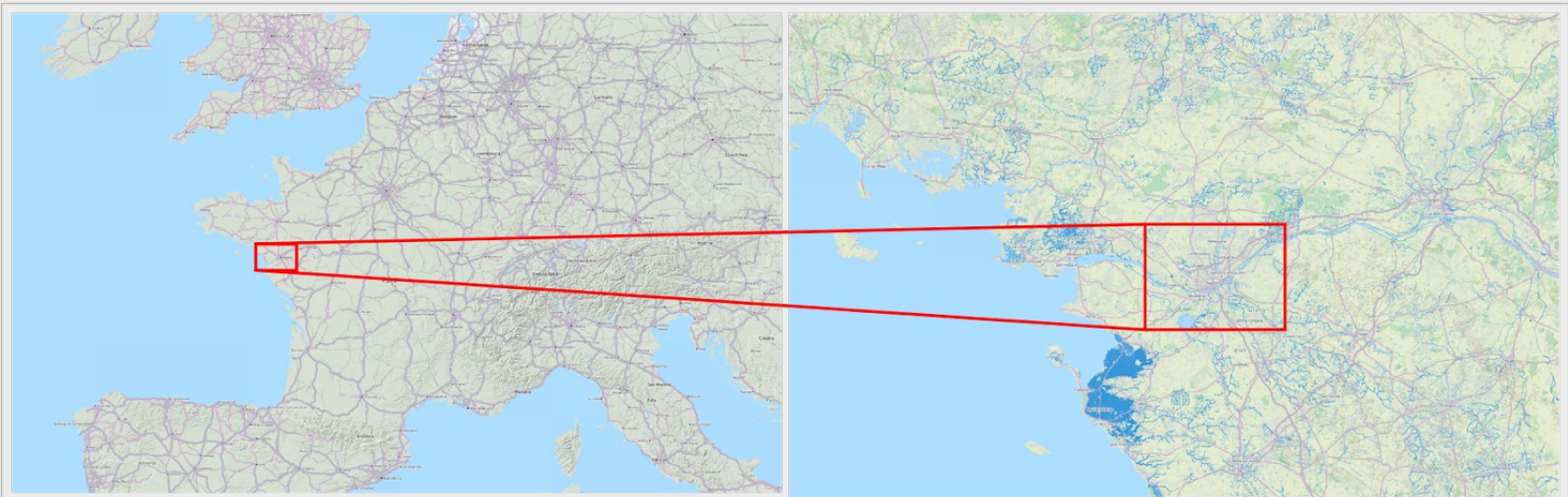
- S3** S2 + Limited Traffic Zone reducing car traffic

Study area



Nantes Métropole

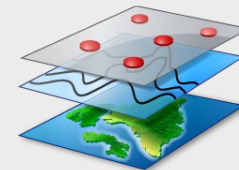
- 24 cities around the city of Nantes
- 590 000 inhabitants
- 524 km²



Source : www.openstreetmap.org - 2014

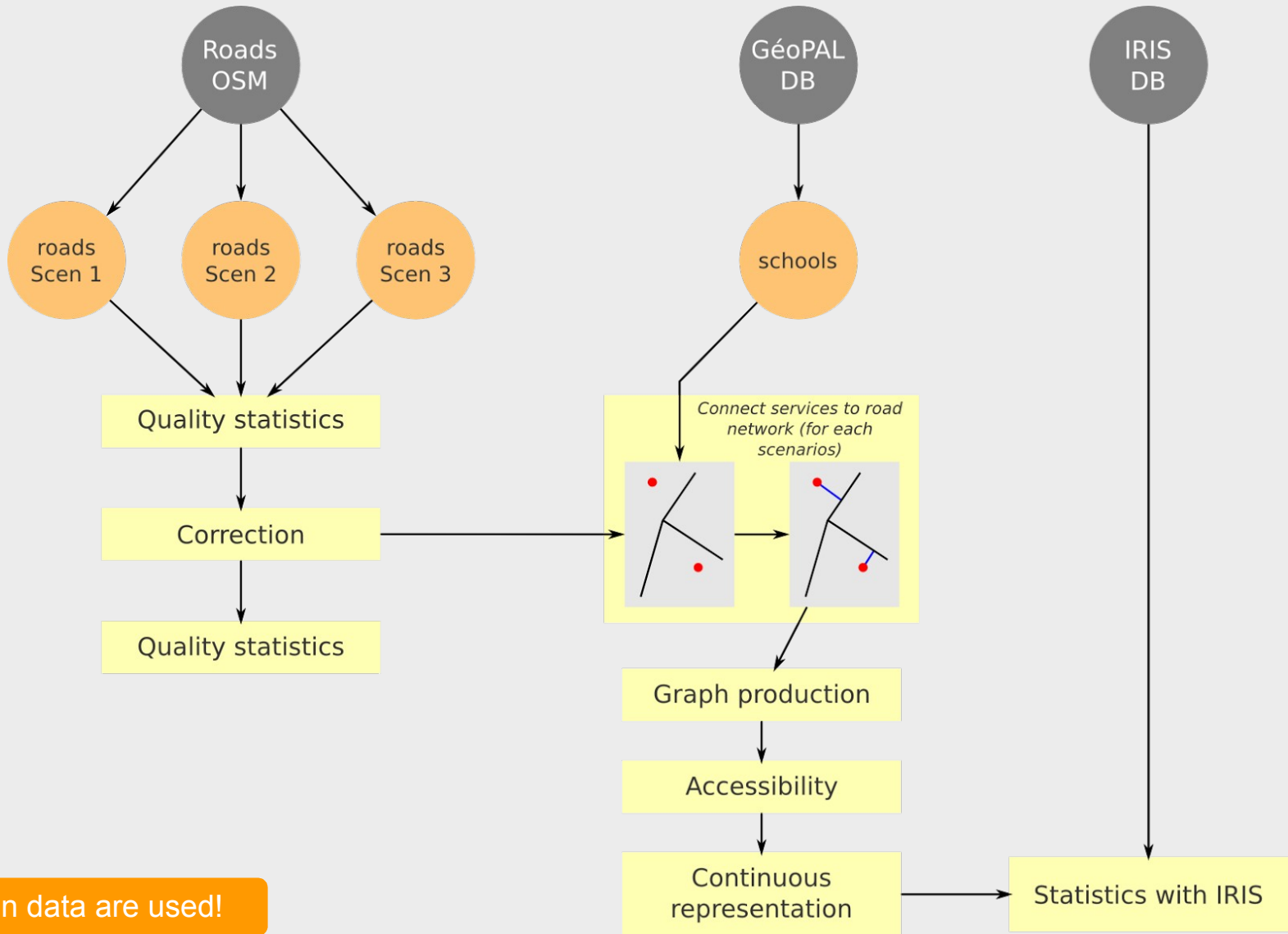
0 5km

Open data



- OSM from GeoFabrik (*snapshot from May 21th, 2014*)
- Schools from the GeoPAL platform, a regional Spatial Data Infrastructure (*downloaded on May 21th, 2014*)
- IRIS squares (200m) from INSEE website (*downloaded on May 29th, 2014*)

Methodology



Only open data are used!

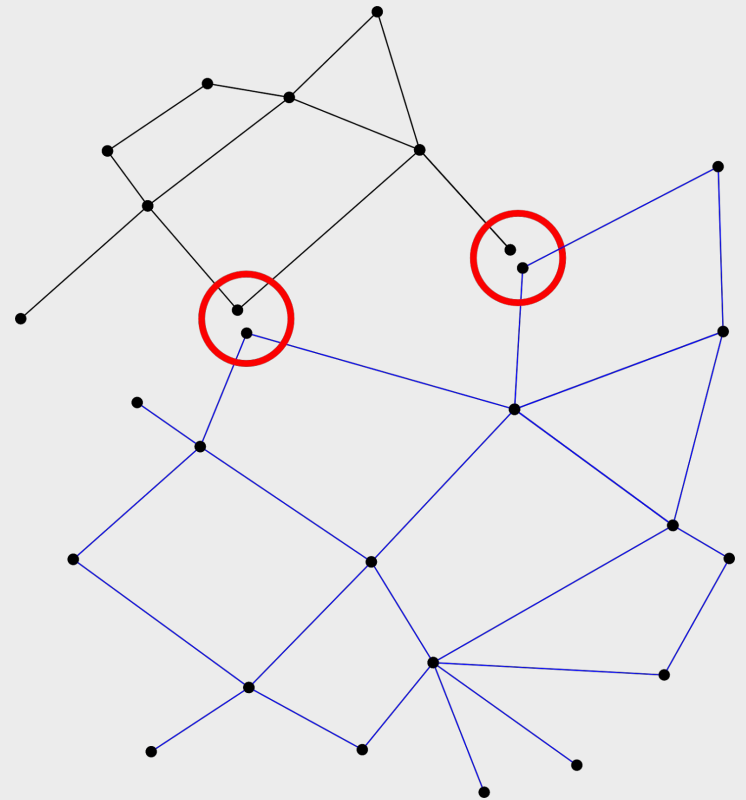
1. Evaluate network connectedness

Goal:

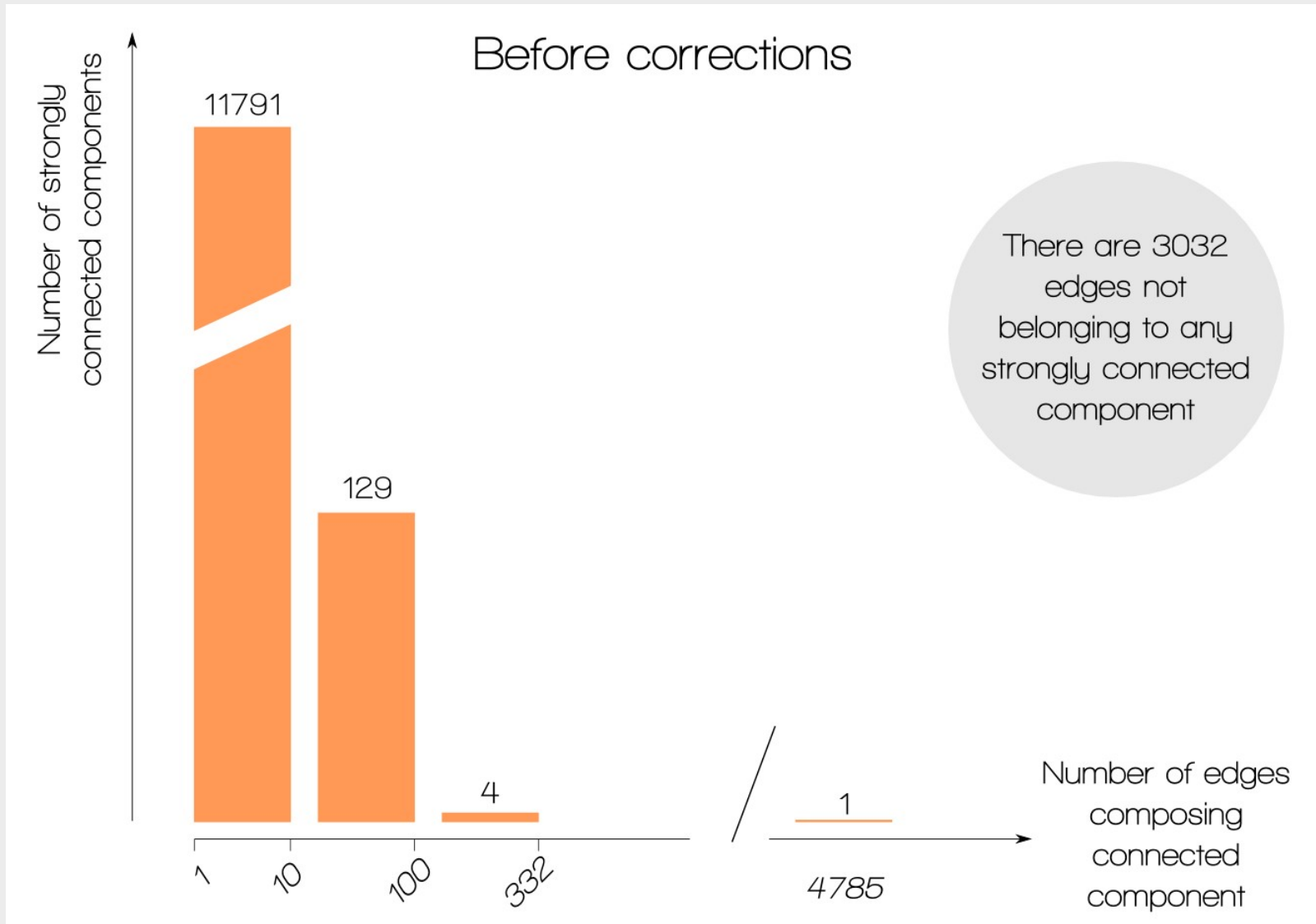
Calculate the number of connected components of the OSM graph, using two H2Network functions:

→ ST_Graph

→ ST_ConnectedComponents



1. Evaluate network connectedness



1. Evaluate network connectedness

Largest component



2. Correct the three road networks

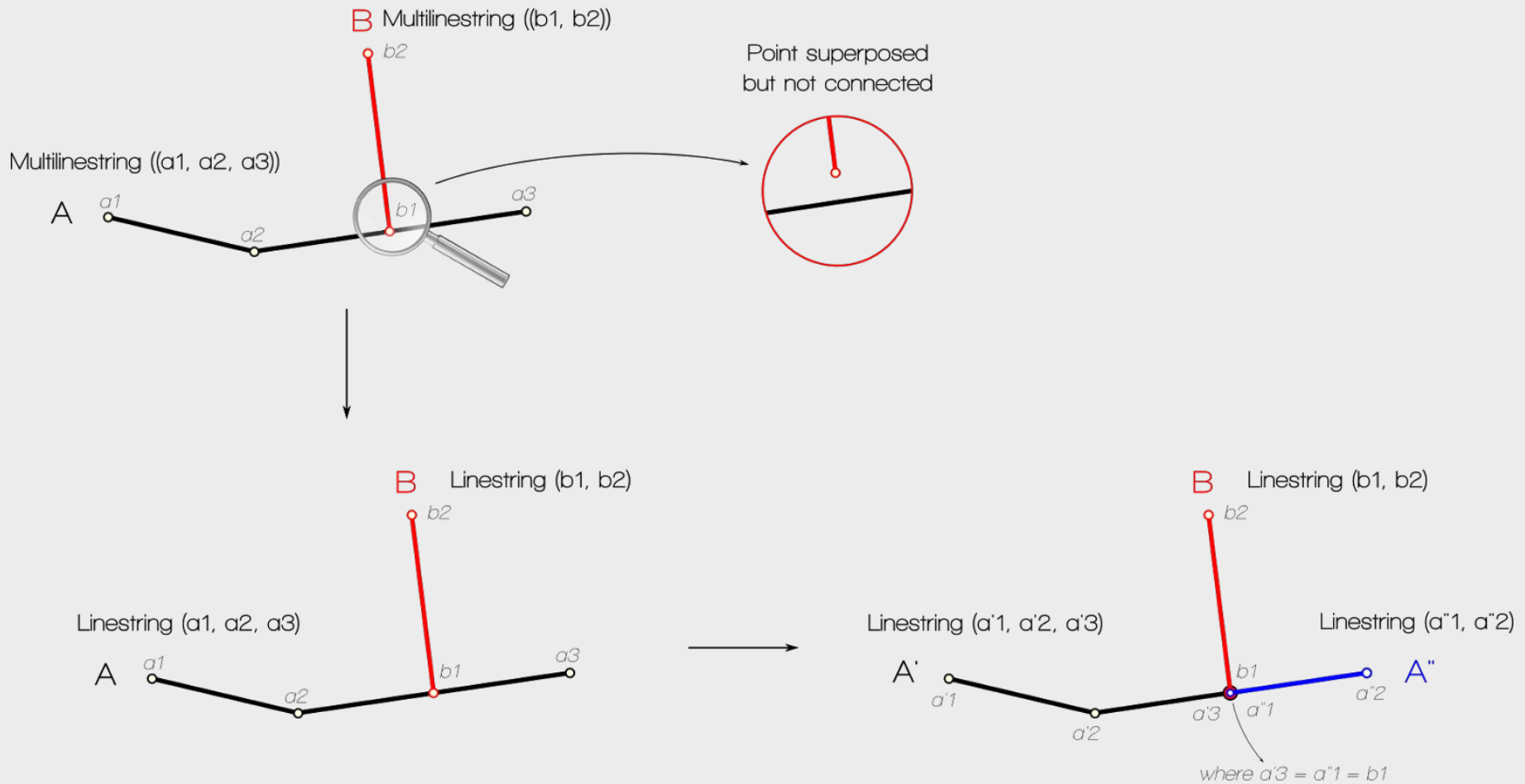
Many edges are not in the largest connected component

Intersections between roads are not necessarily made

→ We propose a 3-step method to split lines and to produce nodes on intersections

2. Correct roads networks

1. Roads from OSM

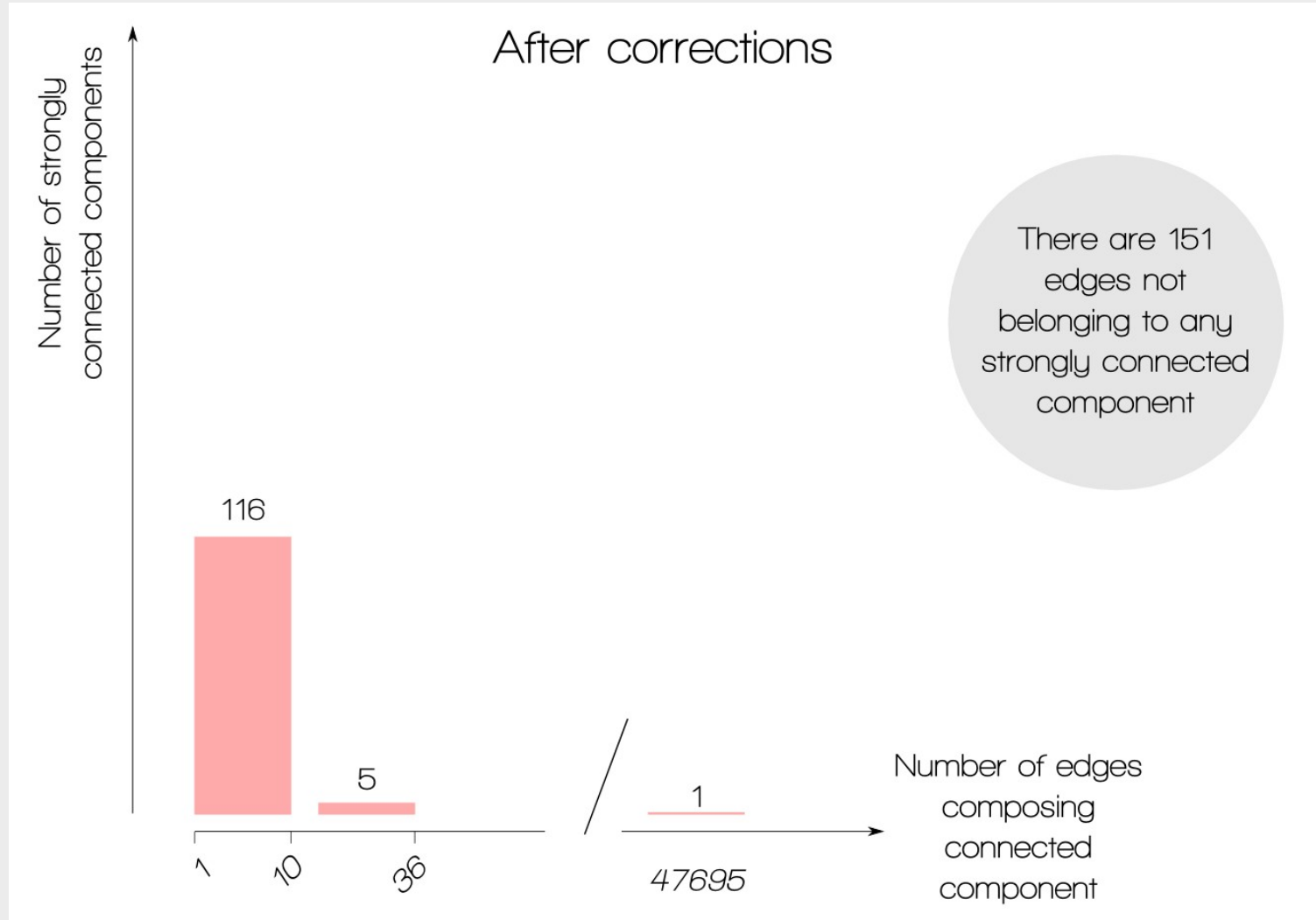


2. Convert to Linestring

3. Intersect Linestrings

3. Compare network connectedness

Same quality analysis, but after the correction



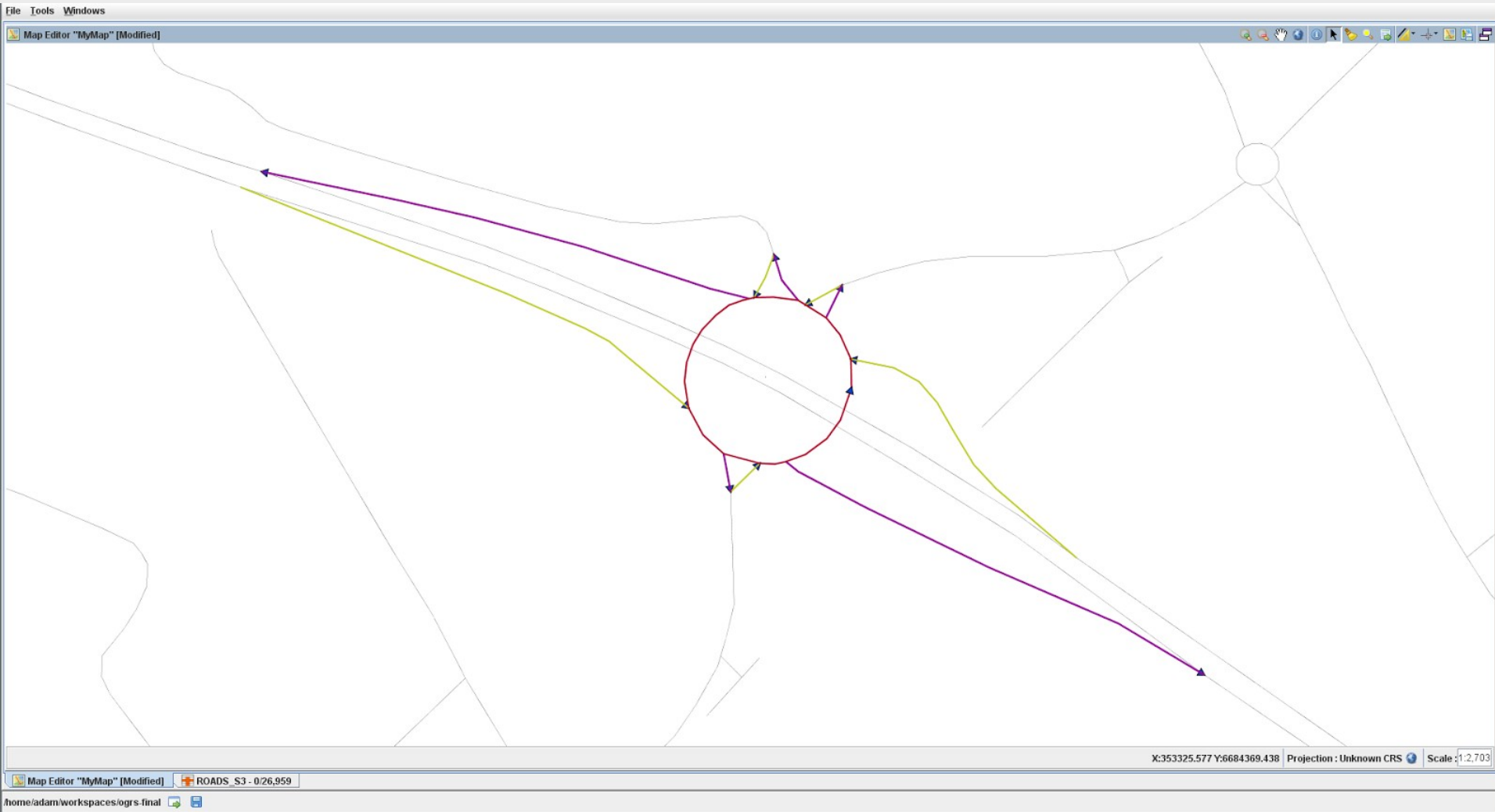
3. Compare network connectedness

Same quality analysis, but after the correction

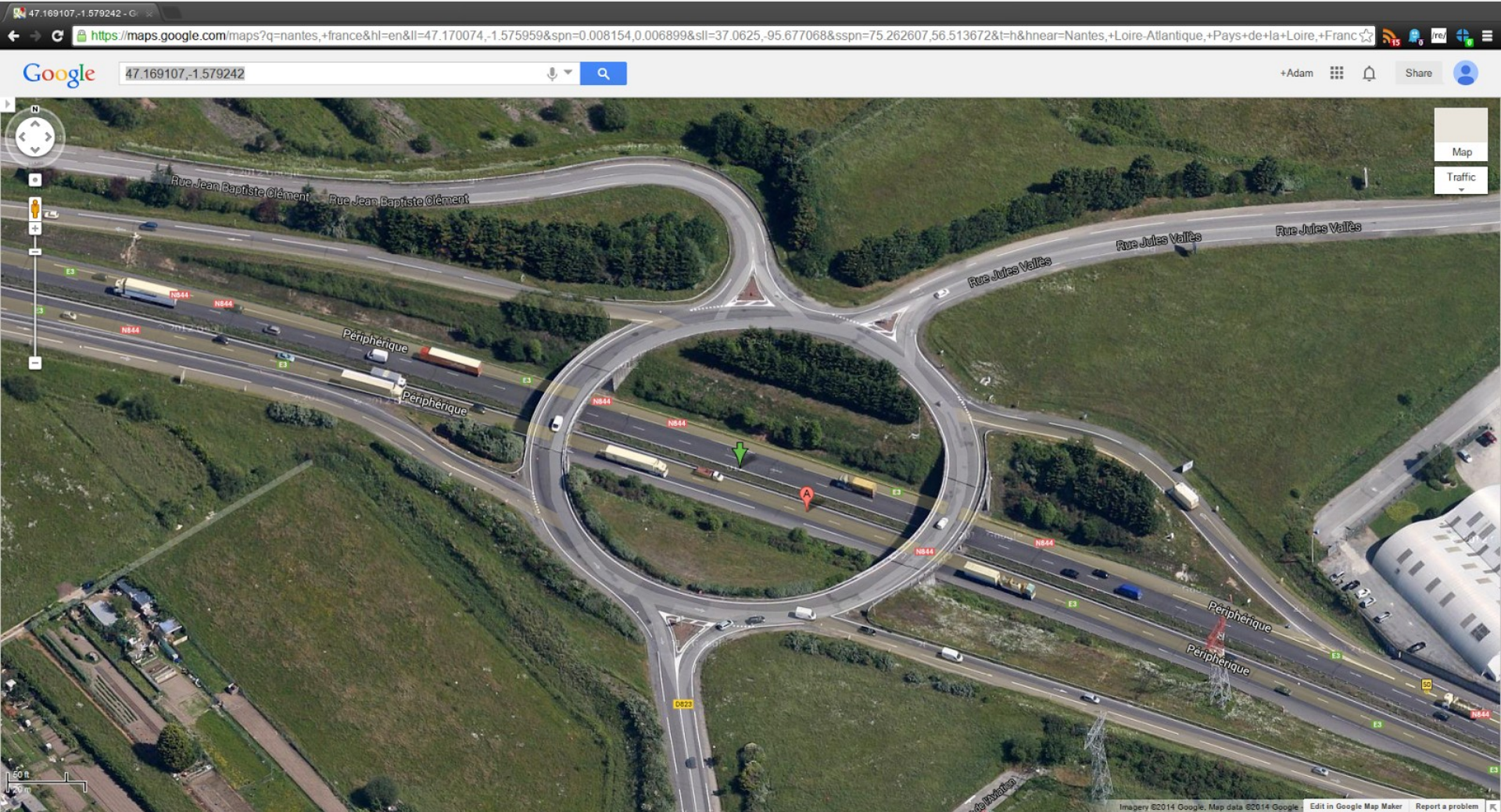
Largest component



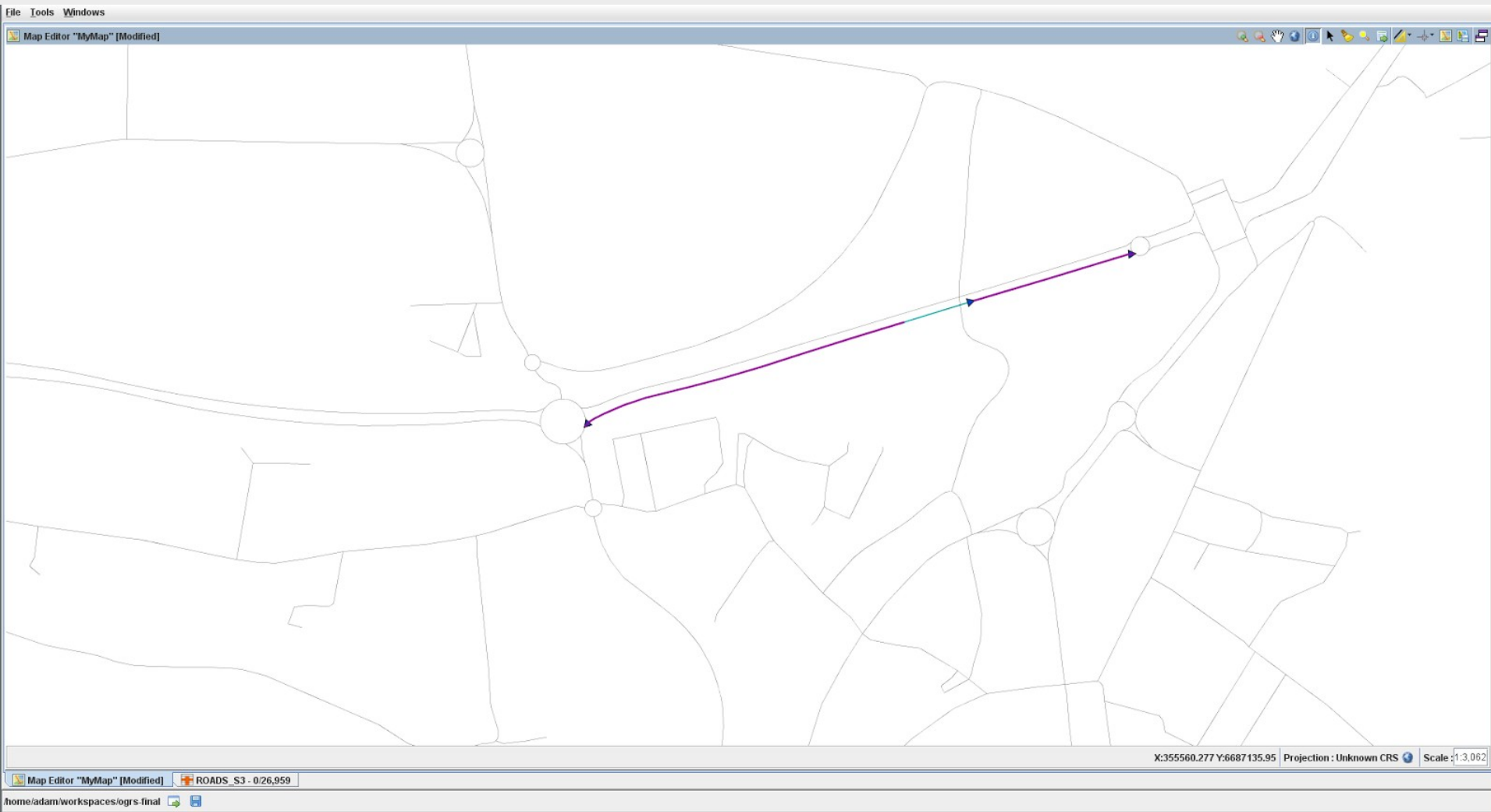
Corrections by hand



Corrections by hand

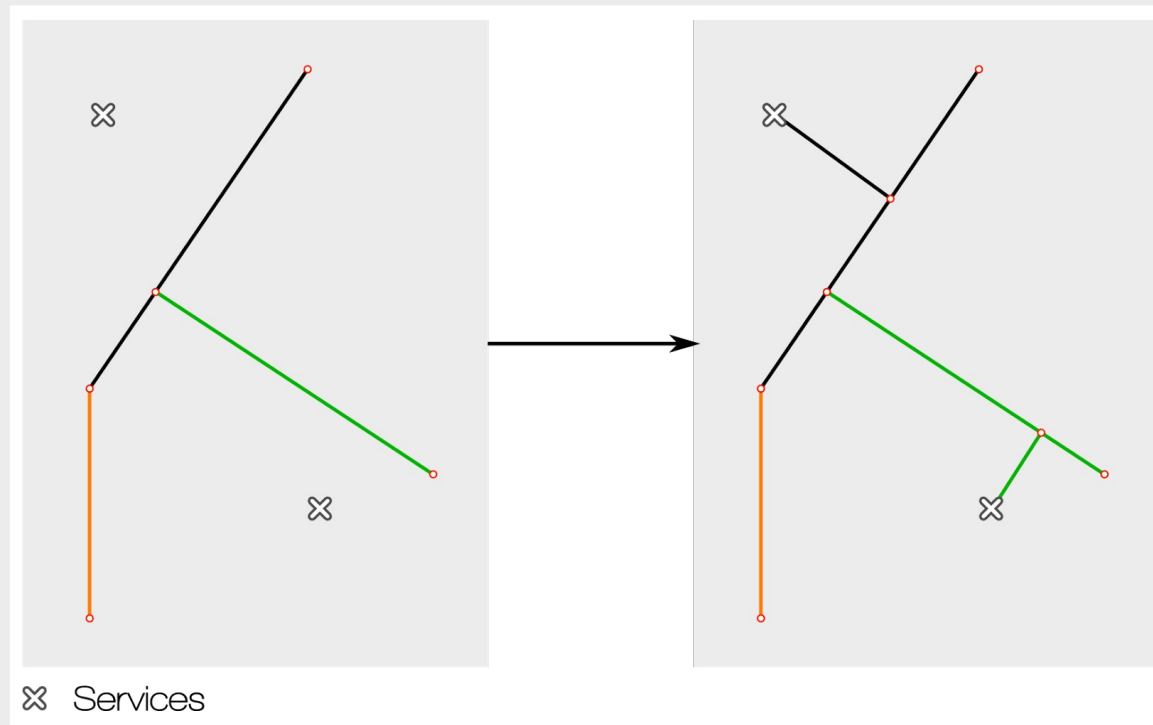


Corrections by hand

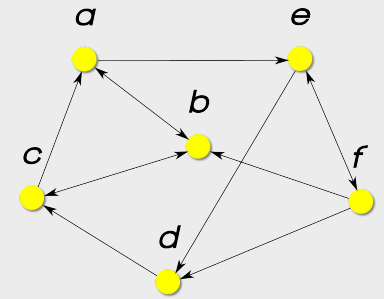


4. Connect services to road network

- Project to nearest road
- Recover ID of intersecting road
- Split road to create new vertex
- Create new ID (PK used by ST_Graph)



5. Produce the graph



Definition: A graph G is an ordered pair

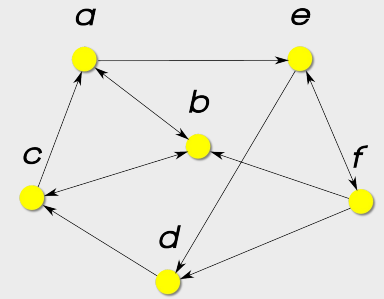
$$G = (V, E)$$

where V is the set of vertices and

$$E = \{ \{v_i, v_j\} : v_k \in V \}$$

is the set of edges

5. Produce the graph



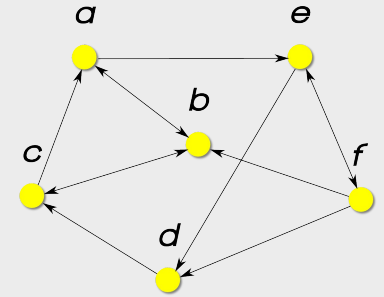
Definition: A *directed graph* has

$$E = \{ (v_i, v_j) : v_k \in V \}$$

Definition: A *weighted graph* is a graph equipped with a function

$$w: E \rightarrow \mathbf{R}$$

Node and Edge tables



```
CALL ST_Graph('ROADS', 'THE_GEOM', 0.1);
```

```
→ ROADS_NODES,  
   ROADS_EDGES
```

Idea: Assign integer IDs to

1. First/last coordinate of each LINESTRING
2. Assign integer IDs to each LINESTRING and its start/end vertices

```
→ ROADS_EDGES table used by other functions
```

5. Produce the graph

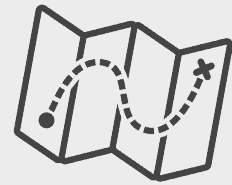
The result

| ROADS_NODES - 0/38 166 | | |
|------------------------|--|--|
| NODE_ID | THE_GEOM | |
| 115 | POINT (352670.5013984775 6688512.875953419) | |
| 116 | POINT (361843.5600283892 6687705.634902644) | |
| 117 | POINT (354771.6045006884 6702988.392643887) | |
| 118 | POINT (364608.1826252287 6695228.01524814) | |
| 119 | POINT (357374.75180166337 6682884.856612413) | |
| 120 | POINT (368476.55798298784 6692998.838625705) | |
| 121 | POINT (358531.08463204664 6695052.130270394) | |
| 122 | POINT (356911.7437649465 6681890.273865681) | |
| 123 | POINT (354733.23448255926 6678635.835068987) | |
| 124 | POINT (363478.88633101864 6693952.914479218) | |
| 125 | POINT (355476.6642451469 6691972.481113828) | |
| 126 | POINT (370540.40509855055 6691030.368320204) | |
| 127 | POINT (356107.2579241391 6694965.302107171) | |
| 128 | POINT (357685.4518804041 6689169.562048571) | |
| 129 | POINT (340875.67792883655 6678998.042912269) | |
| 130 | POINT (339897.3866660958 6697583.8843245795) | |
| 131 | POINT (356780.15750902175 6690246.011628034) | |
| 132 | POINT (357244.2609262098 6690230.064072427) | |
| 133 | POINT (357402.69991451316 6688773.759868432) | |
| 134 | POINT (352615.2022564667 6689382.5314414585) | |
| 135 | POINT (359410.33793443674 6688573.324368541) | |



| ROADS_EDGES - 0/48 129 | | | |
|------------------------|------------|----------|-------|
| EDGE_ID | START_NODE | END_NODE | |
| 1 | 1 | 1 | 30646 |
| 2 | 2 | 2 | 13446 |
| 3 | 3 | 3 | 7060 |
| 4 | 4 | 4 | 26681 |
| 5 | 5 | 5 | 22794 |
| 6 | 6 | 6 | 3419 |
| 7 | 7 | 7 | 9726 |
| 8 | 8 | 8 | 11325 |
| 9 | 9 | 9 | 30962 |
| 10 | 10 | 10 | 1600 |
| 11 | 11 | 11 | 6235 |
| 12 | 12 | 12 | 27599 |
| 13 | 13 | 13 | 29905 |
| 14 | 14 | 14 | 19253 |
| 15 | 15 | 15 | 30222 |
| 16 | 16 | 16 | 32608 |
| 17 | 17 | 17 | 6961 |
| 18 | 18 | 18 | 24700 |
| 19 | 19 | 19 | 18771 |
| 20 | 20 | 20 | 21618 |
| 21 | 21 | 21 | 13854 |
| 22 | 22 | 22 | 14074 |
| 23 | 23 | 23 | 15479 |
| 24 | 24 | 24 | 30383 |
| 25 | 25 | 25 | 26861 |
| 26 | 26 | 26 | 28098 |
| 27 | 27 | 27 | 15300 |
| 28 | 28 | 28 | 30811 |
| 29 | 29 | 29 | 30 |
| 30 | 30 | 30 | 7780 |
| 31 | 31 | 31 | 7592 |
| 32 | 32 | 32 | 32144 |
| 33 | 33 | 33 | 17124 |
| 34 | 34 | 34 | 29567 |

Shortest paths



Shortest paths (SP) are returned as tables containing geometries and several attributes for identifying the path

1.-- Recover edge geometries from ROADS, assuming PK

key column is ID

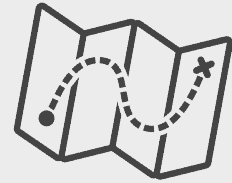
2. CREATE TABLE ROADS_EDGES_GEOM **AS**

3. SELECT A.THE_GEOM, B.*

4. FROM ROADS A, ROADS_EDGES B

5. WHERE A.ID=B.EDGE_ID;

Shortest paths



`ST_ShortestPath('edges', 'o[-eo]', 'w', s, d)`

EDGES Edges table *with geometries*

s Source id

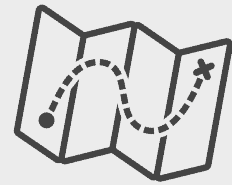
o Global orientation
(directed, reversed or
undirected)

d Destination id

eo Edge orientation
(1 = directed, -1 = reversed, 0
= undirected)

w Edge weight column

Shortest paths

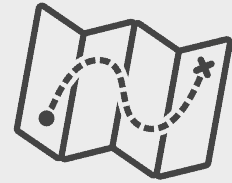


Calculate the shortest path from vertex 34723 (ID in ROADS_NODES) to vertex 15831

1. **CREATE TABLE** SP_34723_15831 **AS**
2. **SELECT ***
3. **FROM** ST_ShortestPath('ROADS_EDGES_GEOM', 'UNDIRECTED', 34723, 15831);

Multiple SPs (common in unweighted undirected)
→ ST_ShortestPath returns them all,
each with a different PATH_ID

Shortest paths



1. -- Suppose `ROADS_EDGES_GEOM` contains a column `EDGE_O` specifying edge
2. -- orientations relative to the order of a `LINestring`'s coordinates:
3. -- * 1 (directed),
4. -- * -1 (reversed),
5. -- * 0 (bidirectional).
6. -- Directed:

7. CREATE TABLE SP_34723_15831_DIR **AS**

8. SELECT *

9. FROM ST_ShortestPath(`'ROADS_EDGES_GEOM'`, `'DIRECTED - EDGE_O'`,

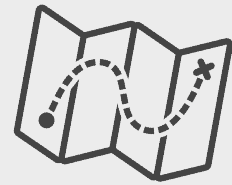
10. 34723, 15831);

11.-- Note: the shortest path from A to B is generally not the same as

12.-- the shortest path from B to A.

13.-- Reduce edge orientations globally: replace `DIRECTED` by `REVERSED`.

Shortest paths



1. -- Suppose `ROADS_EDGES_GEOM` contains a column `WEIGHT` specifying edge weights.

2. -- *Weighted undirected:*

3. **CREATE TABLE** SP_34723_15831_W **AS**

4. **SELECT** *

5. **FROM** ST_ShortestPath(`'ROADS_EDGES_GEOM'`, `'WEIGHT'`, 34723, 15831);

6. -- *Weighted directed:*

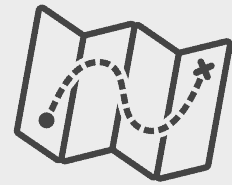
7. **CREATE TABLE** SP_34723_15831_W_DIR **AS**

8. **SELECT** *

9. **FROM** ST_ShortestPath(`'ROADS_EDGES_GEOM'`, `'DIRECTED - EDGE_O'`,

10. `'WEIGHT'`, 34723, 15831);

Distances / Distance matrices



ST_ShortestPathLength('EDGES', 'o[-eo]', 'w', s, d)

One-to-One

ST_ShortestPathLength('EDGES', 'o[-eo]', 'w', s)

One-to-All

ST_ShortestPathLength('EDGES', 'o[-eo]', 'w', 'SDT')

Many-to-Many

ST_ShortestPathLength('EDGES', 'o[-eo]', 'w', s, 'ds')

One-to-Several

| | | | |
|--------------|---|------------|--|
| EDGES | Edges table | s | Source id |
| o | Global orientation (directed, reversed or undirected) | d | Destination id |
| eo | Edge orientation (1 = directed, -1 = reversed, 0 = undirected) | SDT | Source-Destination table (SOURCE, DESTINATION columns required) |
| w | Edge weight column | ds | Comma-separated Destination string ('dest1, dest2, ...') |

6. Measuring accessibility

What is accessibility?

Geographic definition:

The “ease” with which destinations may be reached

Accessibility

- Depends on the transportation network rather than the traveler
- Considers all possible itineraries towards destinations
- Defines a map of possible displacements

(Levy 2003)

6. Measuring accessibility

Graph theoretical definition: Fix a set $D \subset V$ of destinations. Fix a vertex v . Calculate the distance via the road network to each destination d in D . Choose the closest destination.

Implementation: Reverse all edge orientations. Fix a destination d . Calculate the distance to each vertex v . Store d if it is the closest destination found thus far. Repeat for all d in D .

Accessibility

`ST_Accessibility('EDGES', 'o[- eo]',['w'], 'ds')`

`ST_Accessibility('EDGES', 'o[- eo]',['w'], 'dt')`

| | | | |
|--------------------|--|-----------------|--|
| <code>EDGES</code> | Edges table | <code>ds</code> | Comma-separated Destination string ('dest1, dest2, ...') |
| <code>o</code> | Global orientation (directed, reversed or undirected) | <code>dt</code> | Destination table (DESTINATION column required) |
| <code>eo</code> | Edge orientation (1 = directed, -1 = reversed, 0 = undirected) | | |
| <code>w</code> | Edge weight column | | |

6. Measuring accessibility

1. -- Given an input table *DESTS*(destination INT) of vertex IDs:

2. **CREATE TABLE ACC AS**

3. **SELECT ***

4. **FROM** ST_Accessibility('ROADS_EDGES', 'DIRECTED - EDGE_O',

5. **'WEIGHT', 'DESTS');**

6. Measuring accessibility

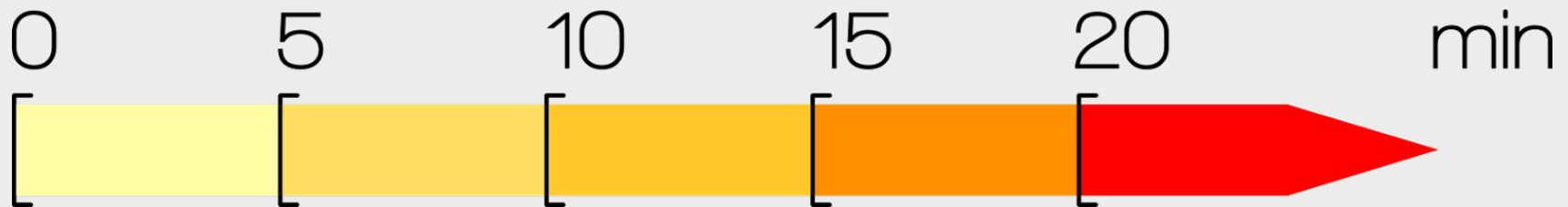
Result →

| ACC - 0/37 428 | | | |
|----------------|----------|--------------|--------------------|
| | SOURCE ▲ | CLOSEST_DEST | DISTANCE |
| 1 | 1 | 31853 | 8056,67111847519 |
| 2 | 2 | 31854 | 6376,185509987808 |
| 3 | 3 | 31854 | 18706,46460767796 |
| 4 | 4 | 31854 | 1636,9495014890324 |
| 5 | 5 | 31855 | 7001,327239000615 |
| 6 | 6 | 31854 | 3068,5624214963564 |
| 7 | 7 | 31855 | 9504,055481371483 |
| 8 | 8 | 31854 | 91914,50691926766 |
| 9 | 9 | 31854 | 4300,071459342103 |
| 10 | 10 | 31854 | 12134,437708286616 |
| 11 | 11 | 31854 | 4408,840821523697 |
| 12 | 12 | 31853 | 9455,22169028394 |
| 13 | 13 | 31854 | 7184,123951352984 |
| 14 | 14 | 31854 | 2294,5976802372434 |
| 15 | 15 | 31853 | 9821,652445023896 |
| 16 | 16 | 31853 | 3605,971838317598 |
| 17 | 17 | 31853 | 12209,791757108502 |
| 18 | 18 | 31854 | 14297,471132164925 |
| 19 | 19 | 31853 | 10419,70732625671 |
| 20 | 20 | 31854 | 6010,471073954411 |
| 21 | 21 | 31853 | 2237,409164102102 |
| 22 | 22 | 31854 | 26637,59686556816 |
| 23 | 23 | 31855 | 4311,777479581588 |
| 24 | 24 | 31854 | 15599,793319455752 |
| 25 | 25 | 31855 | 6288,157553294767 |
| 26 | 26 | 31853 | 8659,108647368825 |
| 27 | 27 | 31854 | 3504,0714341340886 |
| 28 | 28 | 31854 | 1872,2841815807064 |
| 29 | 29 | 31854 | 3673,7603671363045 |
| 30 | 30 | 31854 | 13726,267058266152 |
| 31 | 31 | 31853 | 10770,146563035707 |
| 32 | 32 | 31854 | 33791,69320902031 |
| 33 | 33 | 31853 | 6161,106009461052 |
| 34 | 34 | 31854 | 12323,78024433188 |
| 35 | 35 | 31854 | 25593,036938438927 |
| 36 | 36 | 31853 | 6346,966010365854 |
| 37 | 37 | 31855 | 4110,418791094204 |
| 38 | 38 | 31855 | 16757,00480647751 |
| 39 | 39 | 31853 | 21374,545455985914 |
| 40 | 40 | 31853 | 2296,045778834098 |
| 41 | 41 | 31855 | 6856,995329033835 |
| 42 | 42 | 31854 | 10225,693493655372 |
| 43 | 43 | 31854 | 5543,296326621338 |

7. Continuous representation

Representation: Time contour lines

→ Time intervals based on [study](#)



7. Continuous representation

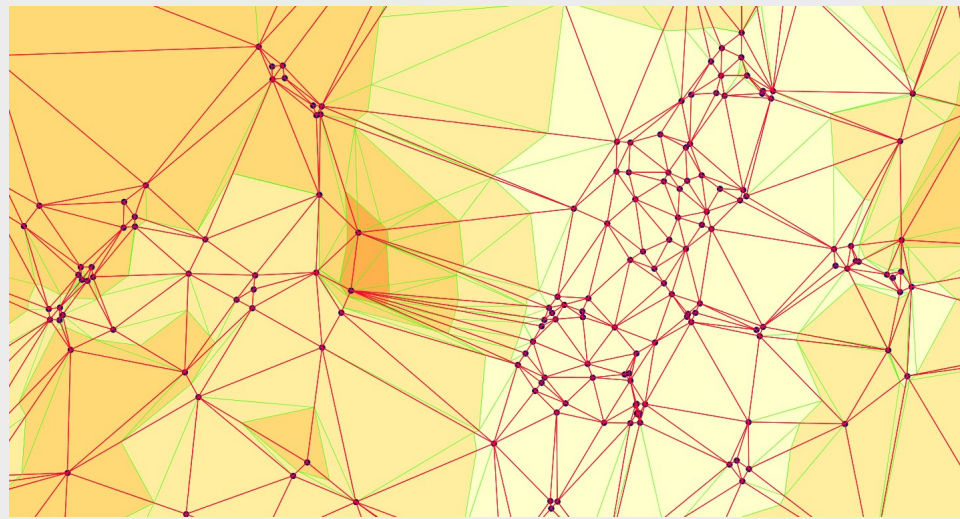
Convert points to contour lines

→ Delaunay constrained triangulation

→ Triangle contouring



Triangulation

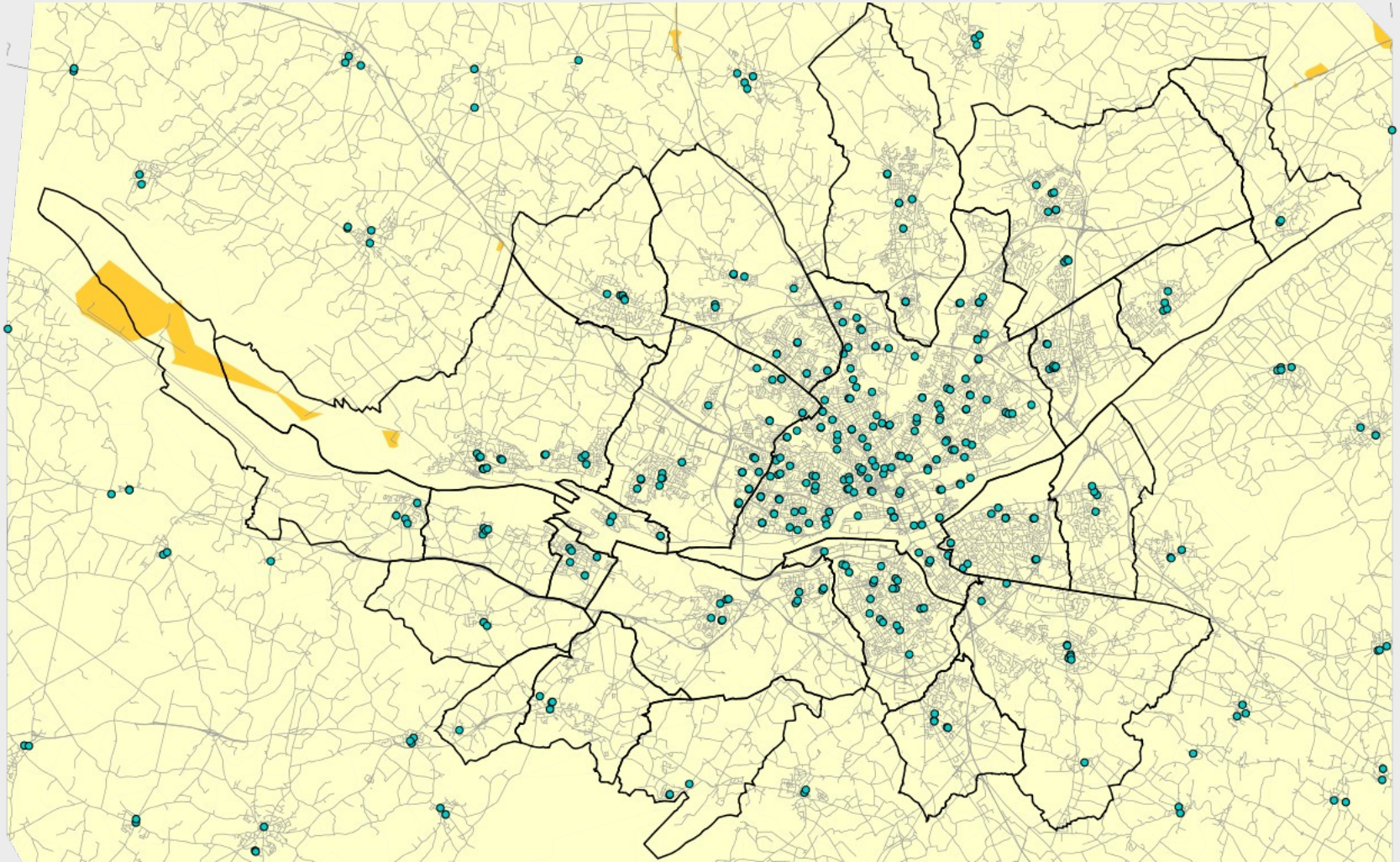
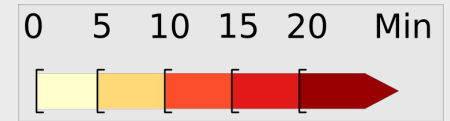


Triangle contouring

→ Use *'time_intervals.se'* style file

Scenario comparison

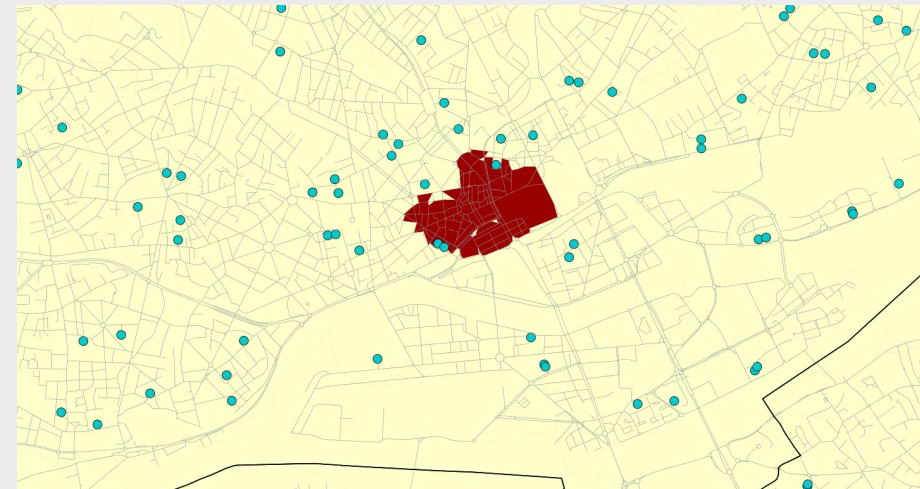
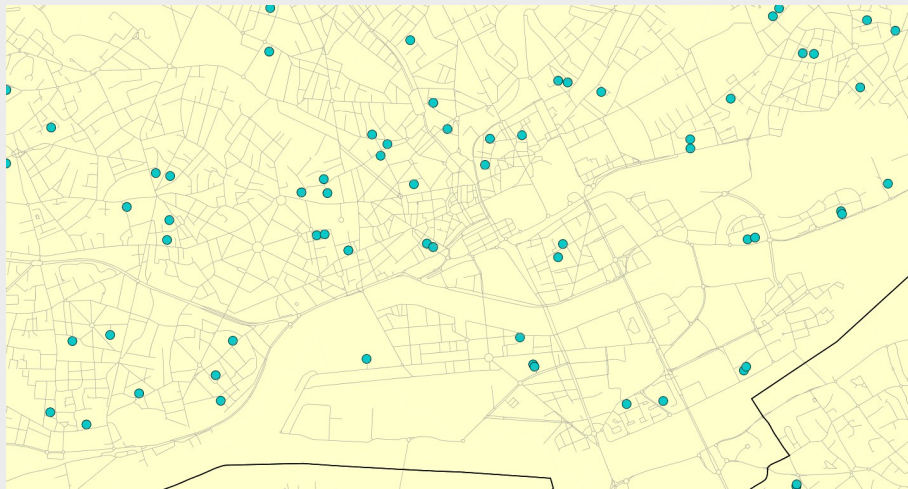
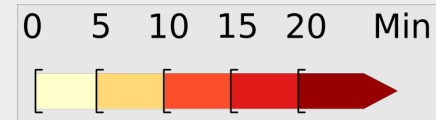
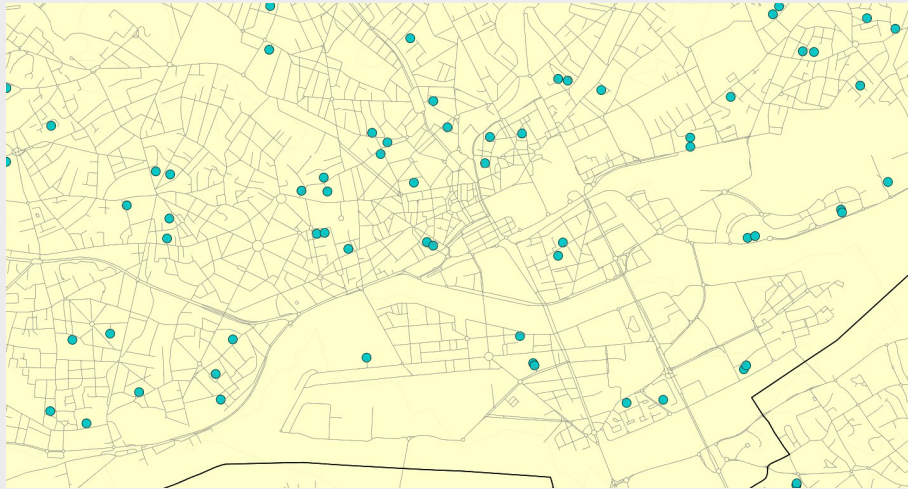
Time map



Scenario comparison

Time maps

Scenario 1



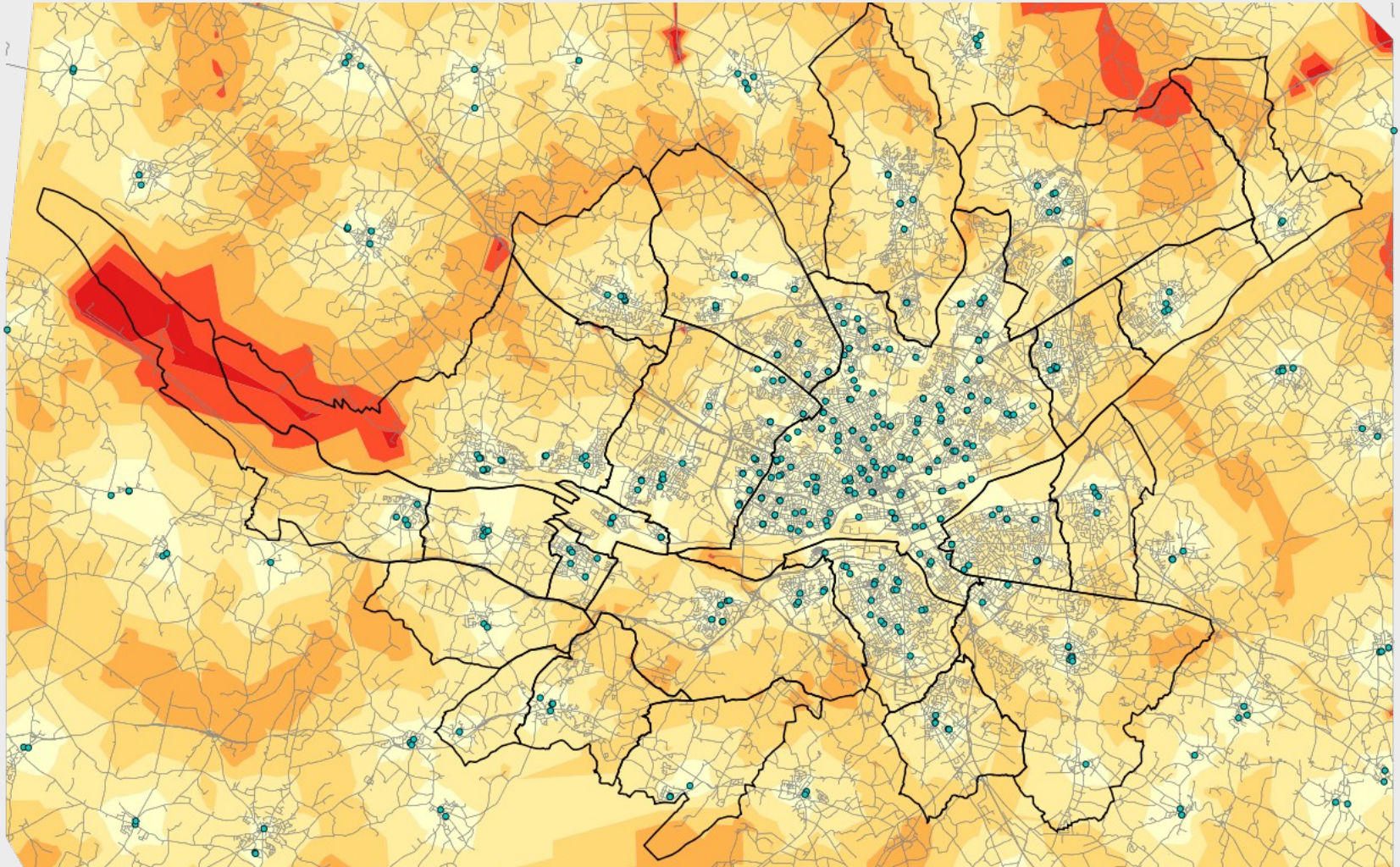
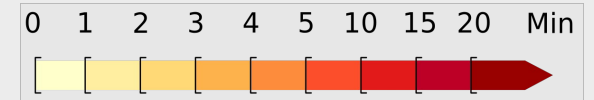
Scenario 2

Scenario 3

Scenario comparison

Time map

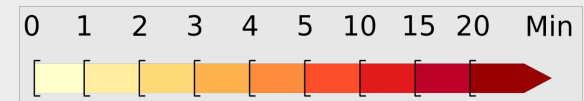
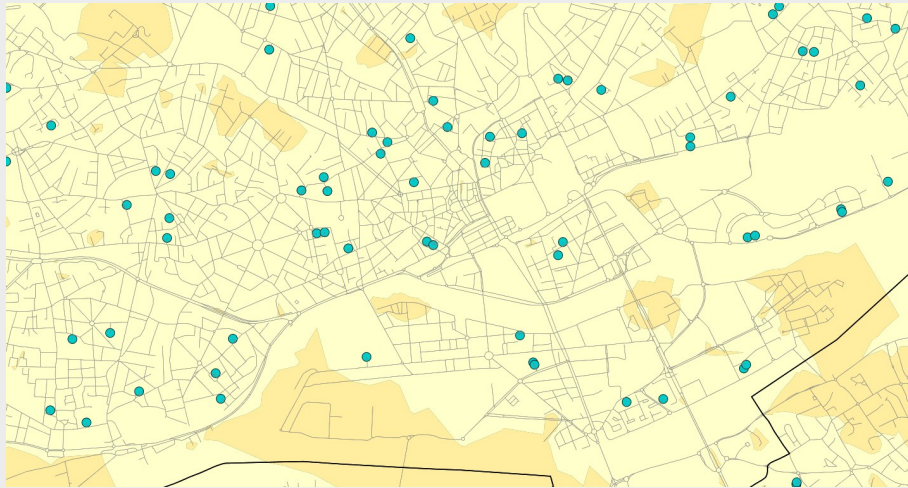
→ *Classification intervals densified*



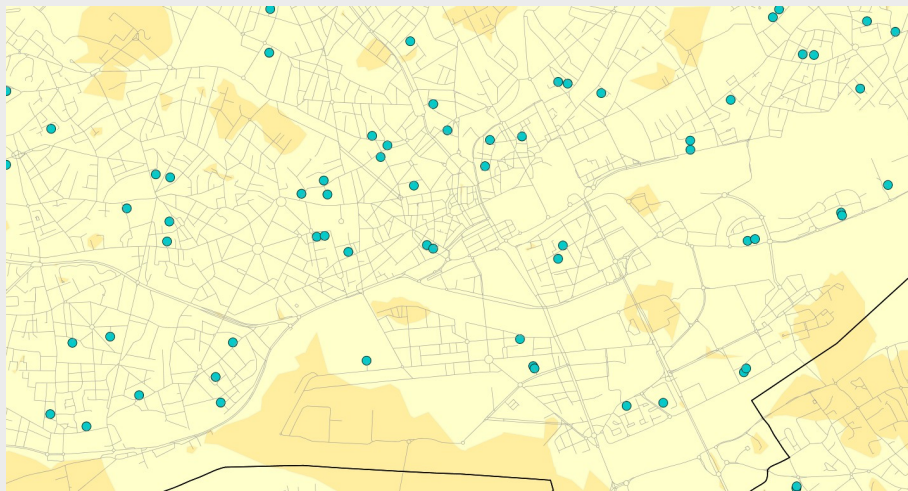
Scenario comparison

Time maps

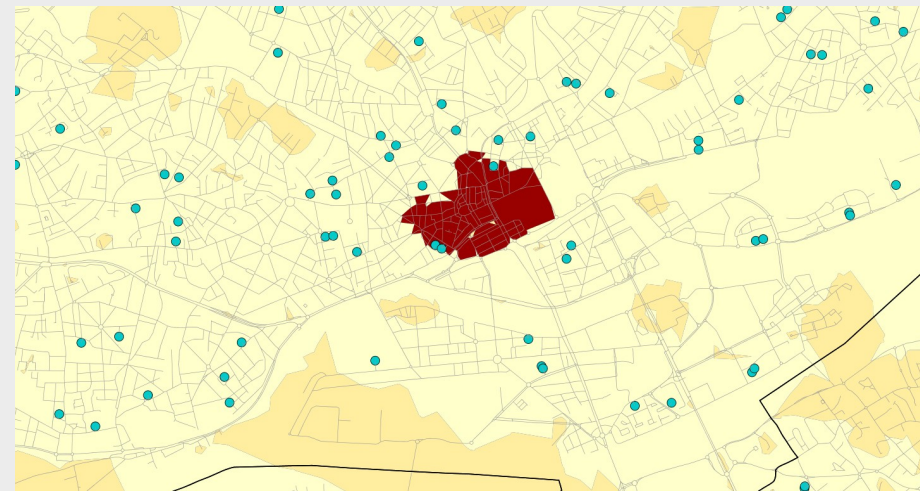
Scenario 1



Scenario 2

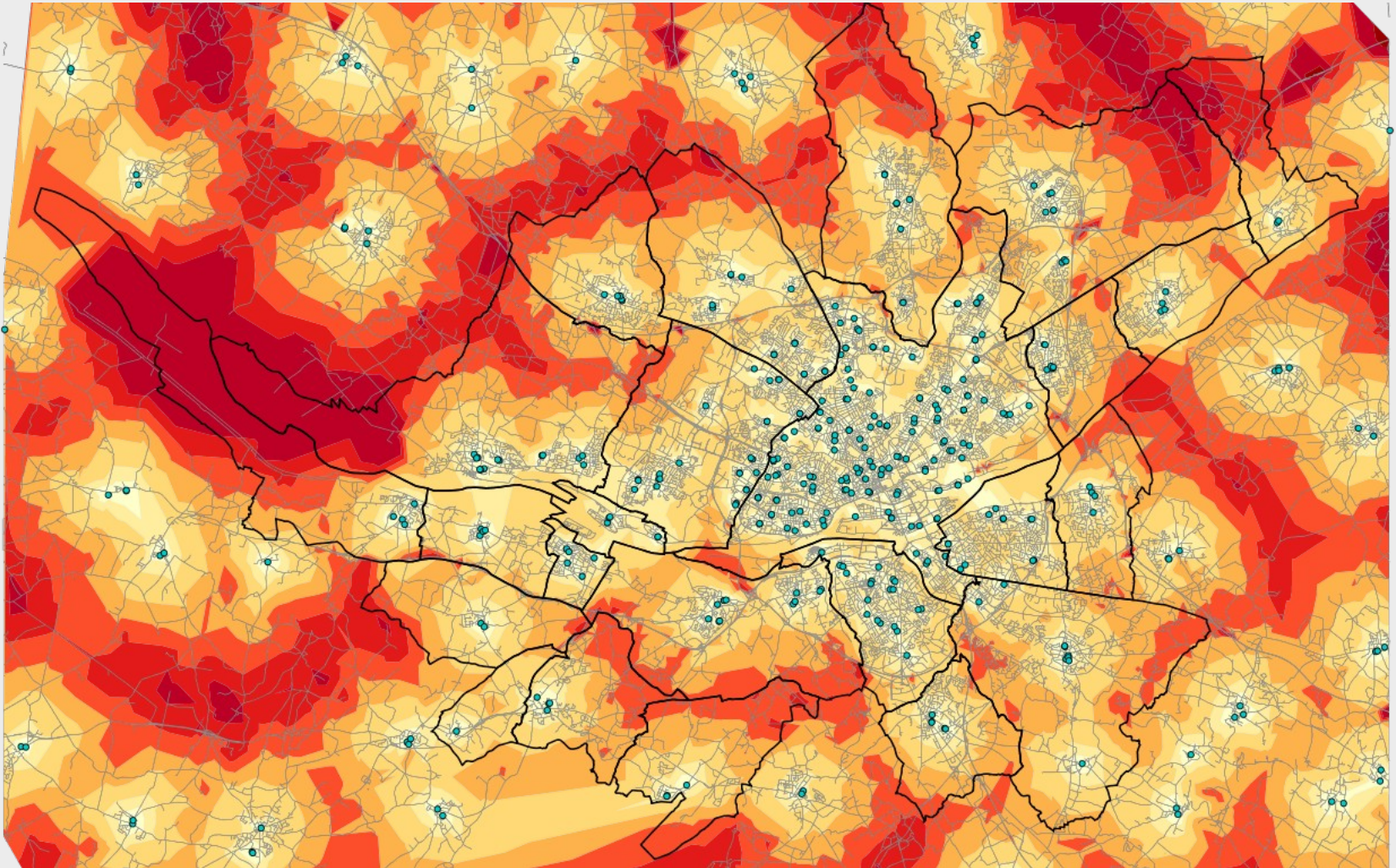
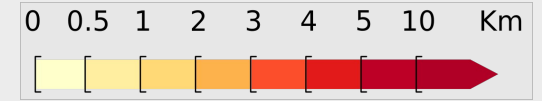


Scenario 3



Scenario comparison

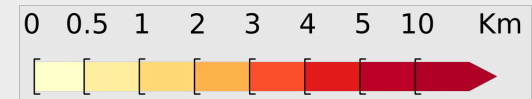
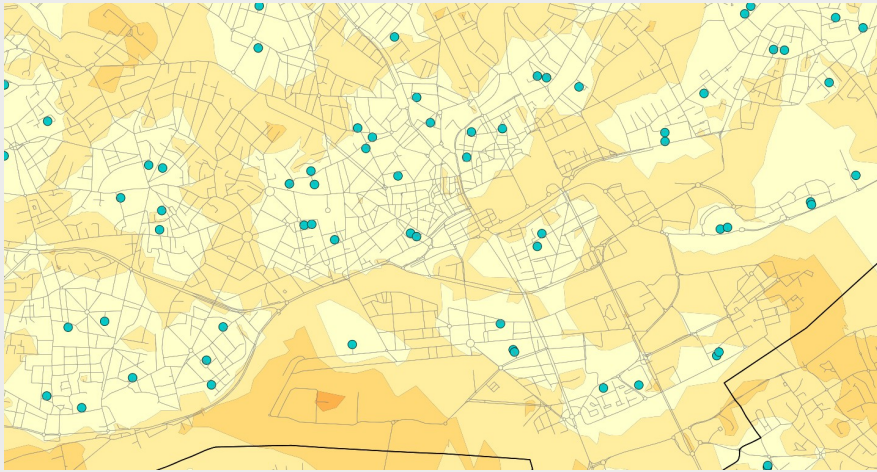
Distance map



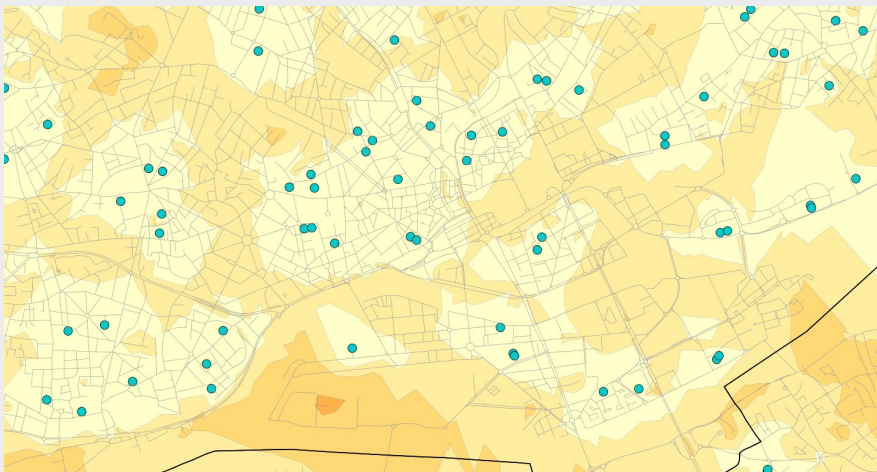
Scenario comparison

Distance maps

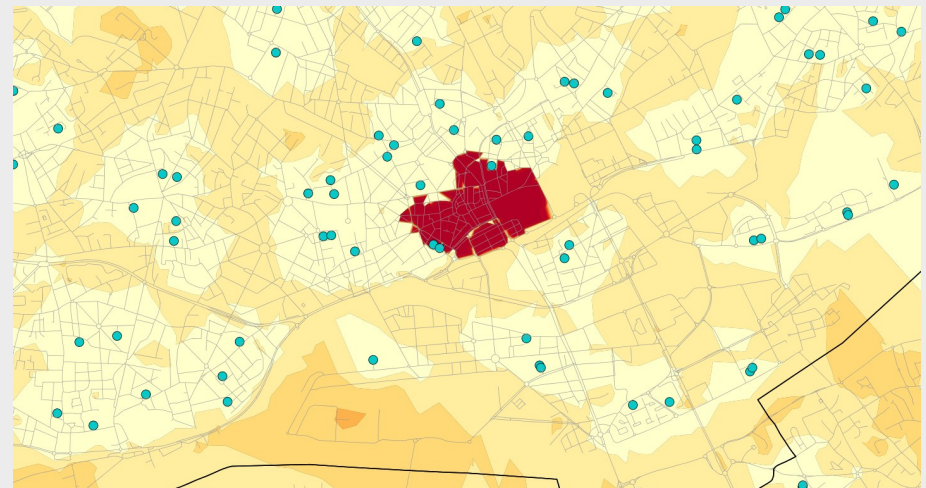
Scenario 1



Scenario 2



Scenario 3



Before concluding

H2GIS in "standalone" mode

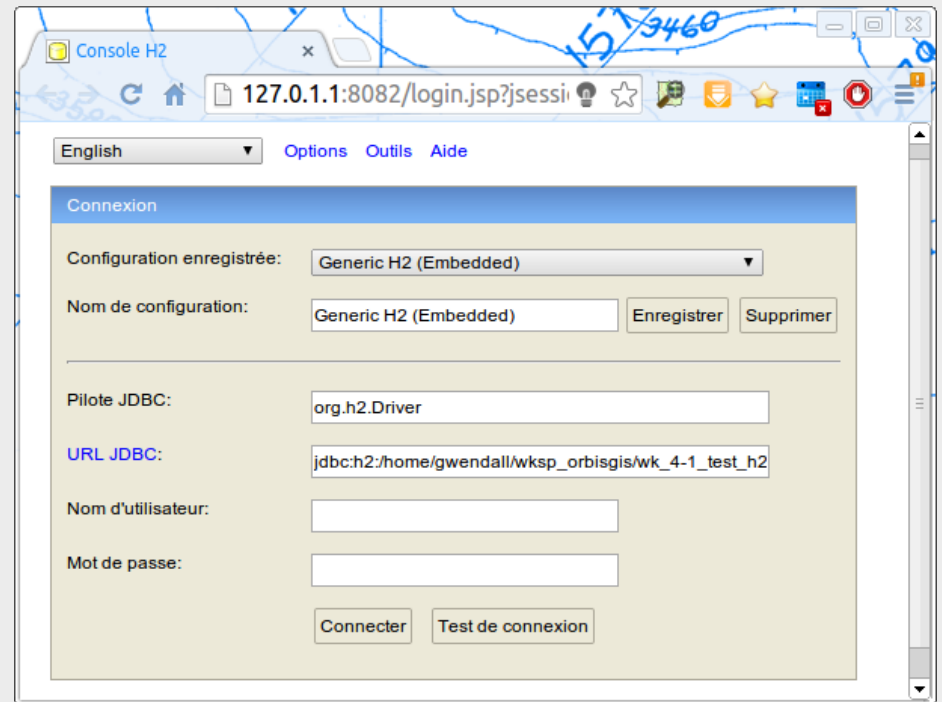
You can use H2GIS in standalone mode in a web browser

Simply extract h2gis-standalone-bin.zip, available on www.h2gis.org

No installation!

Lightweight!
(less than 6MB)

Android compliant



H2GIS in "standalone" mode

jdbc:h2:/home/gwendall/wksp_o

DEPARTEMENT
COMMUNE
SPATIAL_REF_SYS
GEOMETRY_COLUMNS
INFORMATION_SCHEMA
Utilisateurs
H2 1.3.176 (2014-04-05)

Exécuter Run Selected Complètement automatique Effacer Instruction SQL:

Commandes principales

| | |
|---|--|
| ? | Affiche cette page d'aide |
| 📄 | Affiche l'historique des commandes |
| ▶ | Ctrl+Enter Exécute la commande courante |
| 👉 | Shift+Enter Executes the SQL statement defined by the text selection |
| ▶ | Ctrl+Space Complètement automatique |
| 🔌 | Ctrl+Space Déconnexion de la base de données |

Exemple de script SQL

| | |
|---|---|
| Effacer une table si elle existe | DROP TABLE IF EXISTS TEST; |
| Créer une nouvelle table avec les colonnes ID et NAME | CREATE TABLE TEST(ID INT PRIMARY KEY, NAME VARCHAR(255)); |
| Ajouter un nouvel enregistrement | INSERT INTO TEST VALUES(1, 'Hello'); |
| Ajouter un autre enregistrement | INSERT INTO TEST VALUES(2, 'World'); |
| Requêter une table | SELECT * FROM TEST ORDER BY ID; |
| Modifier un enregistrement | UPDATE TEST SET NAME='Hi' WHERE ID=1; |
| Effacer un enregistrement | DELETE FROM TEST WHERE ID=2; |
| Aide | HELP ... |

Ajouter des pilotes de base de données

Des pilotes additionnels peuvent être configurés en déclarant l'emplacement du fichier Jar contenant ces pilotes dans les variables d'environnement H2DRIVERS ou CLASSPATH. Exemple (Windows): Pour ajouter la bibliothèque C:/Programs/hsqldb/lib/hsqldb.jar, définir la valeur de la variable d'environnement H2DRIVERS en C:/Programs/hsqldb/lib/hsqldb.jar.

Conclusion

Conclusion

Using open-source tools and open data we have demonstrated techniques for analyzing road networks.

These analyses are very useful for evaluating UMPs and can help public officials make good decisions.

Future work

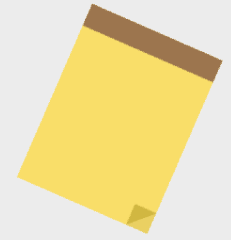


Synchronize data with OSM's servers

Automatize the process and use WPS to provide a web service which is always up-to-date

Measure accessibility relative to public transportation stops and service frequency at each stop

Note



This workshop features practical applications of some of the ideas found in the paper *“H2Network: A tool for understanding the influence of urban mobility plans (UMP) on spatial accessibility”*.

You are welcome to join us tomorrow at 16:00 during the *Urban analysis and applications* session.

Acknowledgments

The OrbisGIS-H2GIS team would like to thank the French [Belgrand-GEBD](#) project for supporting their research, as well as



**Thank you
for your attention**

Questions



Bibliography



Bocher, E., Leduc, T., Moreau, G. & Gonzalez Cortes, F. (2008), “GDMS: an abstraction layer to enhance spatial data infrastructures usability”, in ‘11th AGILE International Conference on Geographic Information Science-AGILE 2008’.

Bocher, E. & Petit, G. (2012), “OrbisGIS: Geographical information system designed by and for research”, Innovative Software Development in GIS pp. 23–66.

Herring, J. (2010), “OpenGIS Implementation Specification for Geographic information - Simple feature access - part 2: SQL option”, Technical report, Open Geospatial Consortium.

Herring, J. (2011), “OpenGIS Implementation Specification for Geographic information - Simple feature access - Part 1: Common architecture”, Technical report, Open Geospatial Consortium.

Lévy, J. & Lussault, M. (2003), “Dictionnaire de la géographie et de l’espace des sociétés”, Belin, Paris.

Newman, M. E. (2003), “The structure and function of complex networks”, SIAM review 45(2), 167–256.

Rodrigue, J.-P., Comtois, C. & Slack, B. (2013), “The geography of transport systems”, Routledge.